НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО

Факультет інформатики та обчислювальної техніки
(назва факультету, інституту)

Кафедра автоматизованих систем обробки інформації і управління
(назва кафедри)

"На правах рукопису"
УДК *004.89*

«До захисту допущено»
Завідувач кафедри

_____ О.А.Павлов
(підпис) (ініціали, прізвище)

“____” _____ 20*18* р.

МАГІСТЕРСЬКА ДИСЕРТАЦІЯ

на здобуття ступеня магістра

за спеціальністю *122 Комп'ютерні науки та інформаційні технології*
(код та назва спеціальності)

спеціалізацією *Інформаційні управляючі системи та технології*
(код та назва спеціалізації)

на тему: *Система аналізу транзакцій блокчейну*

Виконав: студент *VI* курсу групи ІС-61м
(шифр групи)

Данильчук Руслан Костянтинович _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник *доц., к.т.н. Жураковська О.С.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант *к.т.н., доц. Жданова О.Г.* _____
(науковий ступінь, вчене звання, прізвище, ініціали) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Київ – 2018

РЕФЕРАТ

Магістерська дисертація: 89 с., 21 рис., 15 табл., 1 додаток, 36 джерел.

Актуальність. Блокчейн - технологія зберігання інформації, яка знайшла застосування у фінансових операціях. Все частіше люди по всьому світу стикаються з терміном "блокчейн", але далеко не всі розуміють як саме побудований механізм системи. Данна технологія може зробити досить вагомий внесок у розвиток сучасного бізнесу і полегшення життя суспільства. У зв’язку з цим актуальною науковою задачею є оцінка транзакцій блокчейну, що дозволить визначити надійність та можливі перспективи для розвитку певної мережі блокчейну.

Зв'язок роботи з науковими програмами, планами, темами. Робота виконувалась на кафедрі автоматизованих систем обробки інформації та управління Національного технічного університету України «Київський політехнічний інститут ім. Ігоря Сікорського» в рамках теми «Дослідження типових архітектурних шаблонів структур баз даних для широкого кола практичних задач» (№ 0117U000921).

Мета дослідження – спрощення аналізу поведінки та визначення профілю користувача за його адресою в мережі блокчейну.

Для досягнення мети необхідно виконати наступні **завдання**:

- виконати огляд існуючих методів та метрик для оцінювання блоків транзакцій в мережі блокчейну;
- здійснити порівняльний аналіз різних методів та моделей оцінювання блоків транзакцій в мережі блокчейну;
- виконати постановку задачі для аналізу ланцюжків транзакцій;
- розробити алгоритмічне забезпечення для вирішення задачі кластеризації адрес в мережі блокчейну;
- розробити прототип додатку;
- виконати аналіз отриманих результатів.

Об’єкт дослідження – процес аналізу даних ланцюжків блокчейну.

Предмет дослідження – методи аналізу даних ланцюжків блокчейну.

Методи дослідження – методи математичної статистики та інтелектуального аналізу даних.

Наукова новизна отриманих результатів. В представленій роботі було досліджено методи аналізу поведінки користувачів за властивостями їх транзакцій.

На основі дослідження методів аналізу поведінки користувачів за властивостями їх транзакцій здійснено кластеризацію адрес в мережі за часом та об’ємом зроблених транзакцій з метою прогнозування поведінки існуючих та нових користувачів та можливості аналізу мережі.

У зв’язку зі зростанням популярності даної технології та збільшенням кількості способів її застосування, стає необхідним проводити аналіз, який дозволить зменшити ризики під час користування певною мережею блокчейну та виявляти ролі користувачів в ній.

ТЕХНОЛОГІЯ BLOCKCHAIN; БЛОКЧЕЙН; ЗБЕРЕЖЕННЯ ДАНИХ; БЛОК; МАЙНЕР; УЧАСНИКИ; ЗАПИСИ; КЛЮЧ; СКЛАДНІСТЬ МЕРЕЖІ, СКЛАДНІСТЬ ХЕШУВАННЯ

ABSTRACT

Master's Thesis: 89 pages, 21 images, 15 tables, 1 attachments, 36 references.

Relevance Blokchain - technology of storage of information, which has found application in financial transactions. Increasingly, people around the world are faced with the term "blockchain", but not everyone understands exactly how built the mechanism of the system. This technology can make a significant contribution to the development of modern business and to facilitate the life of society. In this regard, the actual scientific task is to evaluate the blocks of blocks, which will determine the reliability and possible prospects for a certain crippling currency.

Connection of research with scientific programs, plans, topics. The work was performed at the the department of Computer-Aided Management And Data Processing Systems (ASOIU) of the National Technical University of Ukraine "Kyiv Polytechnic Institute. Igor Sikorsky" within the frame of the topic « Research of typical architectural templates of database structures for a wide range of practical tasks» (№ 0117U000921).

The purpose and objectives of the study is to simplify the analysis of behavior and define the profile of the user at his address.

To achieve the goal, you must accomplish the following **tasks**:

- perform an overview of existing methods and metrics for evaluating block units in a blockchain network;
- perform a comparative analysis of various methods and models of evaluation of transaction blocks in the blockchain network;
- perform task setting for analysis of transaction chains;
- to develop algorithmic support for solving the problem of clustering of addresses in the blockchain network;
- develop a prototype application;
- perform the analysis of the results.

The object of study – process of data analysis of blockchain.

Subject of research – methods of analysis of data of blockchain.

Methods – methods of mathematical statistics.

Scientific novelty of the research. In the work presented, methods for analyzing user behavior by the properties of their transactions were investigated.

Based on a study of methods for analyzing the behavior of users on the properties of their transactions, clustered addresses in the network by the time and amount of transactions made in order to predict the behavior of existing and new users and the ability to analyze the network.

Due to the growing popularity of this technology and increasing the number of ways to use it, it becomes necessary to conduct an analysis that will reduce risks when using a blockchain network and identify the roles of users in it.

BLOCKCHAIN TECHNOLOGY; BLOCKING; STORAGE OF DATA; BLOCK; MINER; PARTICIPANTS; RECORDS; KEY; COMPATIBILITY OF THE NETWORK, COMPATIBILITY OF HASHING

ЗМІСТ

ВСТУП

Робота присвячена аналізу транзакцій користувачів в мережі блокчейну з метою прогнозування поведінки існуючих та нових користувачів та можливості аналізу мережі. Актуальність задачі обумовлена швидким поширенням застосування технології блокчейну в різних сферах діяльності та важливістю отримувати характеристику користувачів за їх історією діяльності в мережі блокчейну.

Blockchain (з англ. block - блок, chain - ланцюг) - це ланцюжок блоків транзакцій, які зберігаються на комп'ютерах учасників ланцюжка. Кожен наступний блок пов'язаний з попереднім і складається з набору записів. Нові блоки завжди додаються лише в кінець цього ланцюжка.

Ланцюжок даних має три основні принципи:

а) захищеність;

б) розподіленість;

в) відкритість;

Всі учасники блокчейну об'єднуються в комп'ютерну мережу. На кожному сервері зберігається копія всіх даних блоку. Це і є основою надійності blockchain.

Адже, щоб зламати ланцюжок, потрібно отримати доступ до бази даних всіх комп'ютерів мережі.

Записи кожного блоку відкриті для всіх користувачів blockchain. Учасники можуть легко переглянути дані будь-якого з блоків. Зміну інформації в них легко відстежити, так як при цьому змінюється цифровий підпис. Це захищає дані від недобросовісних учасників.

Захист блоків базується на криптографічних ключах. З їх допомогою перевіряється правильність всіх даних системи.

Криптографічний ключ blockchain - це система шифрування, побудована на великій кількості чисел за допомогою алгоритму хеш-функції.

Дані блоків зберігаються на серверах комп'ютерів його учасників.

Всі користувачі мережі поділяються на учасників і майнерів. Учасники створюють записи, а майни перевіряють їх, формують блоки та роздають ці блоки по всій мережі блокчейн. Доки запис не поміщений в блок, він вважається недостовірним.

Учасники зберігають записи на своєму сервері та мають доступ до інших комп'ютерів. Завдяки цьому вони можуть обмінюватися даними з іншими користувачами мережі.

Так як в системі блокчейн можуть знаходитися, як добропорядні, так і зловмисні учасники, відповідно розповсюджуються справжні і підроблені дані.

Блокчейн не потребує посередників, банків, контролерів та державних органів. Всі учасники мають рівні права, можуть робити все, що їм завгодно, в тому числі безуспішно намагатися обманути інших.

Всі дані, що з'являються в блоках відкриті (користувачі бачать їх) і зашифровані (користувачі не знають, кому вони належать).

Приклад запису в мережі блокчейн: "Користувач з ключем К отримав у кредит телефон з ключем S".

Кожен користувач може мати декілька різних ключів. Тобто, навіть знаючи ключ власника телефону, не можна дізнатися про наявність у нього штрафу за порушення правил дорожнього руху.

Кожен блок в blockchain має заголовок і тіло. Тіло блоку - це просто перелік записів у ньому. Заголовок блоку має більш складну структуру.

Заголовки пов'язані між собою ключами. Кожен блок містить ключ від попереднього блоку. Це означає, що в ключі блоку закодовані записи цього блоку та попередніх. Такий принцип забезпечує незмінність та захист системи блокчейн. Навіть при незначній зміні ключа блоку система покаже невідповідність, адже це вимагає зміни ключів всіх наступних блоків.

Таким чином, всі користувачі мережі захищені від обману та дій зловмисників.

Якщо подивитися на адресу в блоці і відповідні їм транзакції у блокчейні, шукати попередній блок, то починають з'являтися окремі кластери адрес, які

демонструють дивно схожі шаблони транзакцій і поведінку. Кожен з них зазвичай відноситься до певного виду користувачів.

Ґрунтуючись на шаблонах, можна зробити висновки про адреси, які можуть бути представлені наступними кластерами:

- адреси, пов'язані з біржами: централізовані біржі криптовалюти зазвичай мають тисячі «гарячих рахунків», які використовуються як взаємозамінні для управління обмінними операціями і транзакціями клієнтів;
- адреси «ботів» або загублені: адреси, які або відображають «автоматичну» модель транзакції (наприклад, повторювані, аналогічні транзакції для одного і того ж одержувача) або є явно одноразовими, які мають тільки одну вхідну і вихідну транзакцію;
- адреси, які належать людям і «інші»: зведена група адрес, які зазвичай відображають нестандартні моделі проведення транзакцій.

Застосовуючи це, ми можемо отримати більш чітке уявлення про розмір мережі, проаналізувавши, чи дійсно даний токен використовується звичайними людьми або це ботнет.

1 ОГЛЯД ВІДОМИХ АЛГОРИТМІВ АНАЛІЗУ ТРАНЗАКЦІЙ БЛОКЧЕЙНУ

1.1 Огляд існуючих метрик та показників блокчейн

В даний час питання надійності систем набувають все більшого значення, це формує нові вимоги до технологій обробки і зберігання даних. Зупинимося докладніше на аналізі технології блокчейн (blockchain або block chain - ланцюжок блоків транзакцій).

Мережа блокчейн – це набір записів (вихідні транзакції або UTXO в блокчен-термінології).

Розглянемо такі розширені функції транзакцій blockchain, як BIP141 та BIP091.

Передача блоків проявляється через транзакції, які містять:

- один або кілька входів, кожен з яких стосується дійсного UTXO і містить інформацію про аутентифікацію, яка дозволяє це зробити, витрачаючи відповідні віртуальні кошти;
- один або кілька виходів, кожен з яких містить специфікацію щойно створених UTXO.

Після завершення транзакції посилання на UTXO видаляються. Сукупна вартість транзакційних виходів зазвичай трохи менша, ніж сукупна вартість входів в систему блокчейн. Різниця (комісія за транзакцію) виплачується в вузлі блокчейн, який включає транзакцію до блоку.

Для того, щоб витрачати віртуальні кошти, наприклад біткойни, власник, як правило, повинен представити в мережі загальнодоступну інформацію про аутентифікацію, що майно належить йому.

Можна вважати, що:

- кожна адреса блокчейн контролюється одним реальним суб'єктом господарювання. Таким чином, ми цілком ігноруємо рідкісні випадки,

коли використовується багатопасивна адреса для спільного володіння віртуальними коштами, а не для багатофакторних аутентифікацій;

– один об'єкт може контролювати більше однієї адреси.

У міру того як будь-який blockchain вдосконалюється і розвивається, безперебійне функціонування децентралізованої мережі стає першою та необхідною умовою її життєздатності. Отже існує 11 найбільш значущих показників для визначення надійності та швидкості роботи мережі [1].

1.1.1 Час реплікації даних

Обмін інформацією в мережі blockchain не відбувається миттєво. Шкала реплікації даних показує, як швидко транзакція досягає 50 відсотків всіх учасників пірингової мережі (тобто скільки часу пройшло між тим, як транзакція або блок потрапили в мережу і тим моментом, коли більшість вузлів мережі отримало це оновлення).

1.1.2 Списки стартових серверів

У випадку першого підключення до системи, клієнт має дізнатись адреси інших вузлів. Списки відомих вузлів мережі (стартових серверів) використовують всі клієнти для визначення робочих вузлів в мережі, з якими клієнт буде встановлювати з'єднання при початку роботи. Списки відомих вузлів підтримують і поширюють волонтери, використовуючи різні методи, для того щоб вперше приєднуються до мережі вузли отримали чітке уявлення про працюючі в даний момент в мережі вузли.

Списки надають відомості тільки про підключені до мережі і доступні вузли. Метрика їх доступності відображає результати спроб підключення до вузлів, що містяться в різних бутстрап-списках:

$$p = \frac{\sum_{i=0}^{N} x_i}{N}, \tag{1.1}$$

де $x_i = 1$, якщо спроба підключення до і-го вузла була вдалою, та $x_i = 0$ – якщо ні; N – кількість спроб підключення.

Вона показує, наскільки легко підключитися до мережі новий вузел, який ніколи раніше цього не робив. Чим ближче до 100%, тим більше шанс того, що це вийде з першого разу.

1.1.3 Швидкість хешування мережі

Метрика, що демонструє кількість терахешей в секунду, які мережа генерує за різні часові інтервали (1 терахеш дорівнює 1,000 гігахешей).

Ця серія графіків, спроектована розробником Пітером Вюйле, відображає «складність хешування» [1].

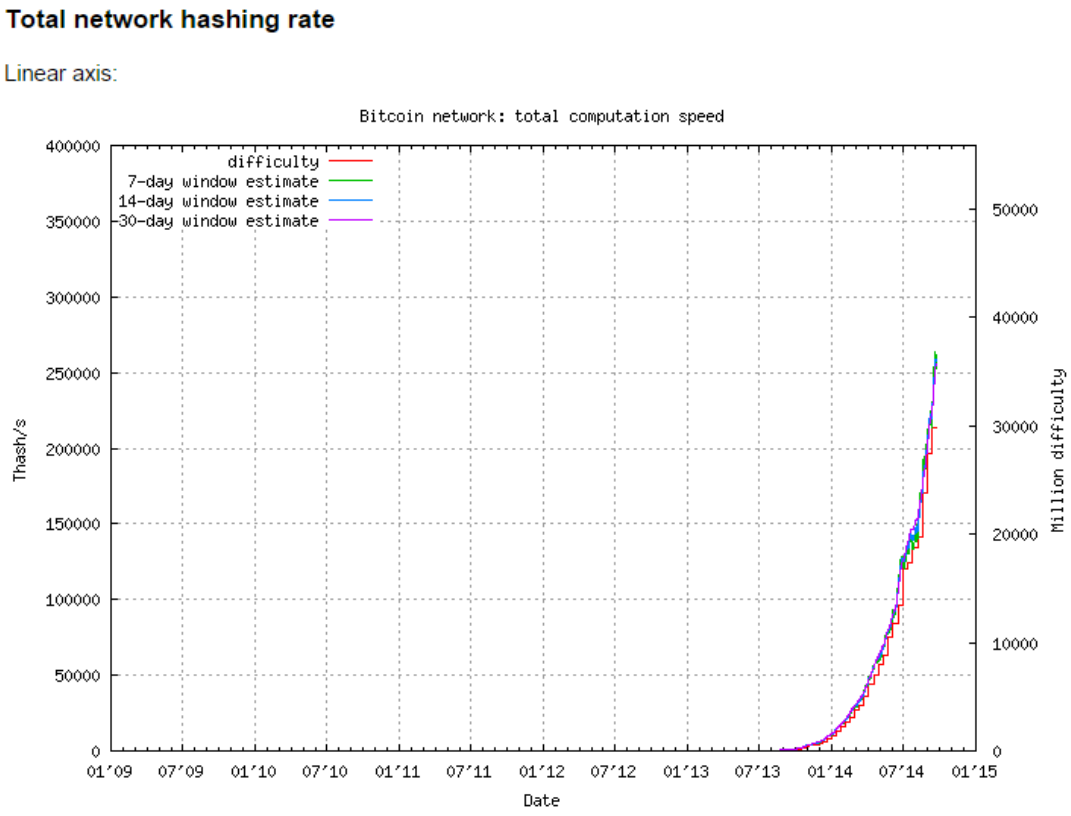

Рисунок 1.1 – Графік швидкості хешування мережі bitcoin

Складність хешування відображає, наскільки важко знайти новий блок в порівнянні з початковою складністю, з якою був випущений генезис-блок (початкова

складність приймається за одиницю). Показник складності автоматично переглядається кожні 2,016 блоків (приблизно раз в 2 тижні).

1.1.4 Розподіл хеш-активності

Хеш-активність – діяльність вузла мережі, що спрямована на обрахування значення хеш-функцій для блоків у ланюцжку транзакцій.

Цей показник важливий, оскільки цілісність мережі залежить від того, щоб окреме джерело майнінговой потужності не контролював постійно понад 50% хешінговой активності.

Кругова діаграма від Organ Ofcorti демонструє розподіл хеш-активності між найбільшими майнінговимі пулами за останній тиждень [1].

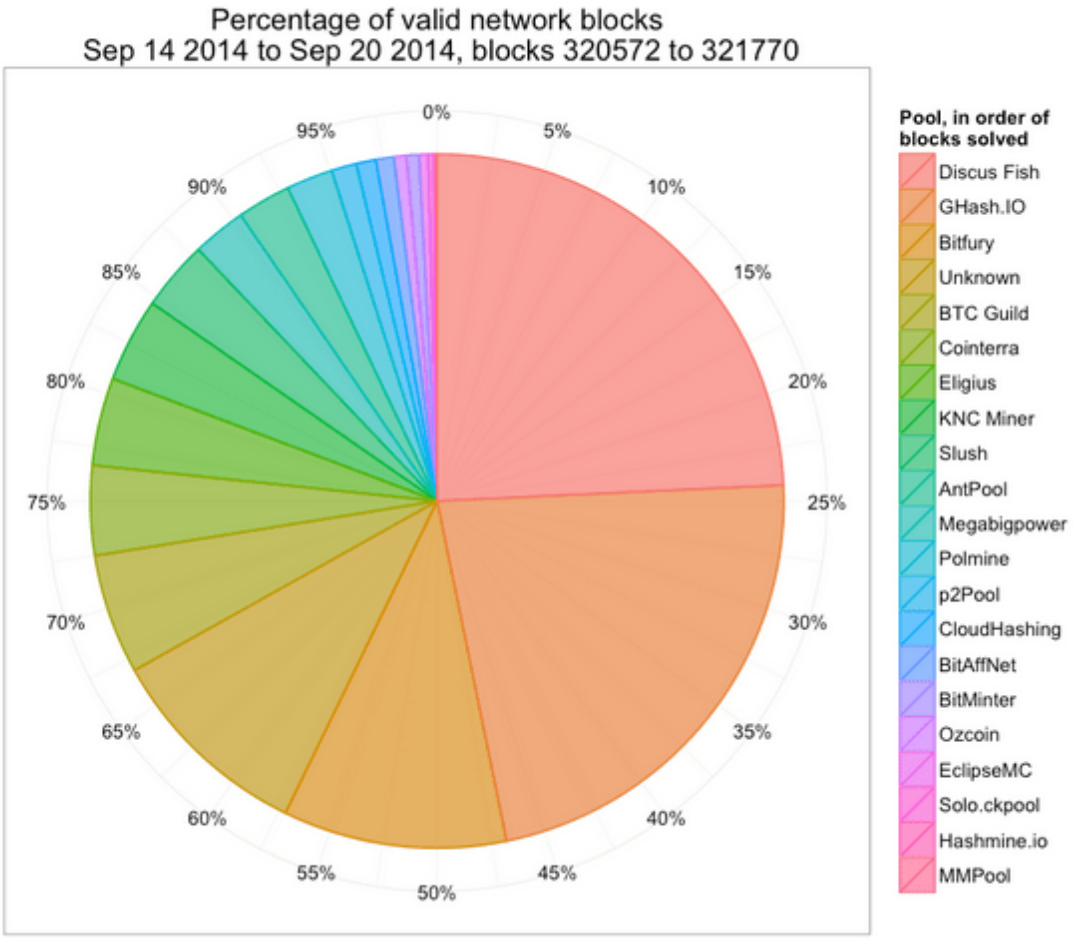

Рисунок 1.2 – Розподіл хеш-активності між найбільшими майнінговими пулами у мережі bitcoin

Таблиця знайдених хеш відображає всі статистичні показники, які можна отримати з кількості блоків, знайдених користувачем за один тиждень. Авторство блоків, як правило, трактується за версією першоджерела, наприклад, за повідомленням на сайті конкретного майнінгового пулу, який заявив про знахідку, іноді за непрямими даними, таким як підписи на Coinbase або відомі відкриті ключі.

Дані, зібрані з інших джерел, можуть виявитися недостовірними і не враховувати окремих блоків, завершення яких майнери вважали за краще зберегти в таємниці, що, безсумнівно, відіб'ється на статистичних показниках рівня хешінговой активності.

1.1.5 Ознаки его-Майнінгу

«Селфіш Майнінг» (его-Майнінг) - стратегія, при якій коаліція особливо егоїстичних майнерів здатна нашкодити мережі, навіть якщо вони не мають 51% потужності. Однак така «атака» має абсолютно чітку сигнатуру - ознаки, за якими его-майнерів можна вирахувати. Розроблена Coinometrics метрика показує поширеність такої моделі поведінки серед майнерів в даний момент [2].

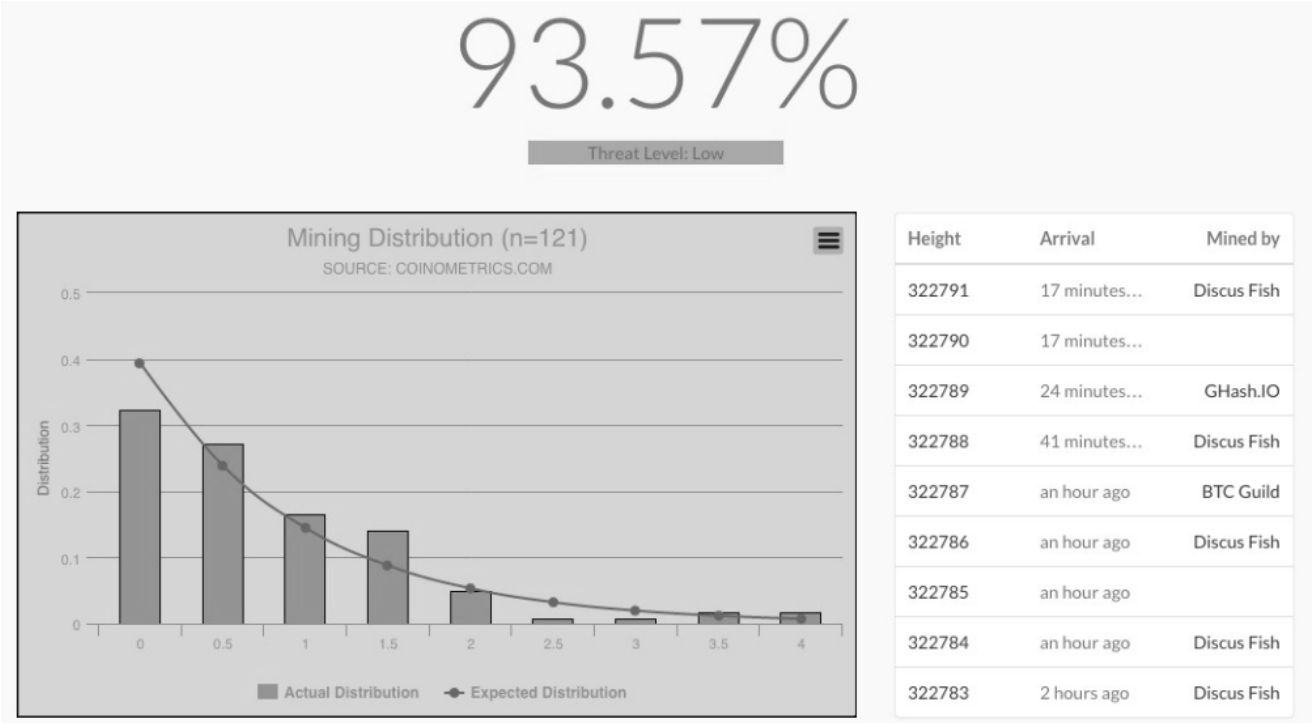

Рисунок 1.3 – Результат роботи Coinometrics

Протокол технології блокчейн передбачає, що як тільки майнери виявляють новий блок, про це слід повідомити всіх учасників мережі.

Его-Майнер не виконують цього припису: коли вони знаходять блок, вони зберігають цей факт в таємниці і приступають до пошуків нового. Коли блоків в ланцюжку стає кілька, вони розкривають карти, виявляючи блоки, які, як інші вважали, все ще знаходяться в розшуку .

Чим менше правдоподібні тимчасові показники, які демонструють лідери списку майнерів до завершення блоків, тим більша ймовірність того, що вони використовують стратегію «селфіш Майнінг». В прикладі (рисунок 1.3), метрика говорить про те, що з імовірністю 94% его-Майнінг не відбувається.

1.1.6 Завислі блоки

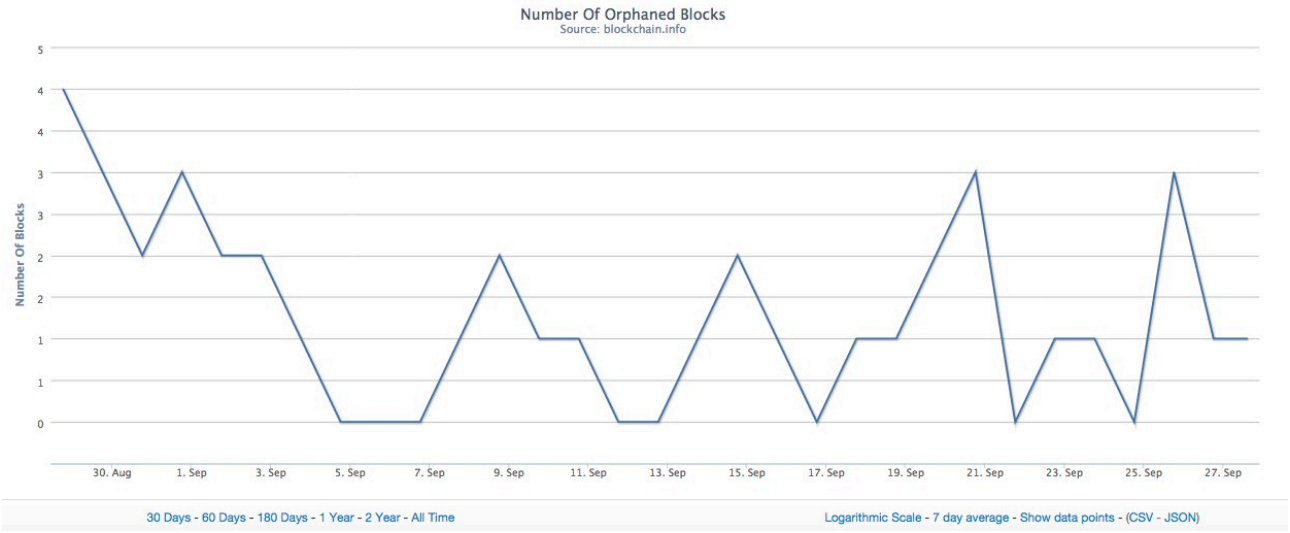

Рисунок 1.4 – Графік кількості завислих блоків в залежності від часу

Завислі блоки - це валідні блоки, які не є частиною основного блокчейну. Вони можуть з'явитися природним чином, наприклад, якщо два майнери завершили один і той же блок одночасно, або виникнути в результаті хакерської атаки, якщо зломщик, що володіє знаннями в хешування, намагається повернути транзакції назад [3].

Завислі блоки вважаються нормальним явищем в біткойн-співтоваристві. Однак їх поява може призводити до такого ефекту. Користувач може спостерігати, як транзакція «повисає», отримавши тільки одне підтвердження, а потім, в новій

оновленої версії блокчейну, кількість підтверджень по тій же транзакції знову повертається до нуля, якщо реалізована нова ланцюжок, довша, без участі першої транзакції.

1.1.7 Моніторинг подвійних витрат

У блокчейні передбачена система контролю за подвійними витратами, що працює в реальному часі: вона застосовувалася при останніх 500 000 транзакціях, з використанням 10 хвилинного кешування. Систему можна використовувати для оповіщення користувачів про можливі атаки зловмисників в мережі.

1.1.8 Непідтверджені транзакції

Блокчейн підтримує постійно оновлюваний список транзакцій, які чекають щоб їх оформили в новий блок. Система контролю відображає загальне число непідтверджених транзакцій та їх обсяг, вимірюваний в кілобайтах.

1.1.9 Середній час підтвердження транзакцій

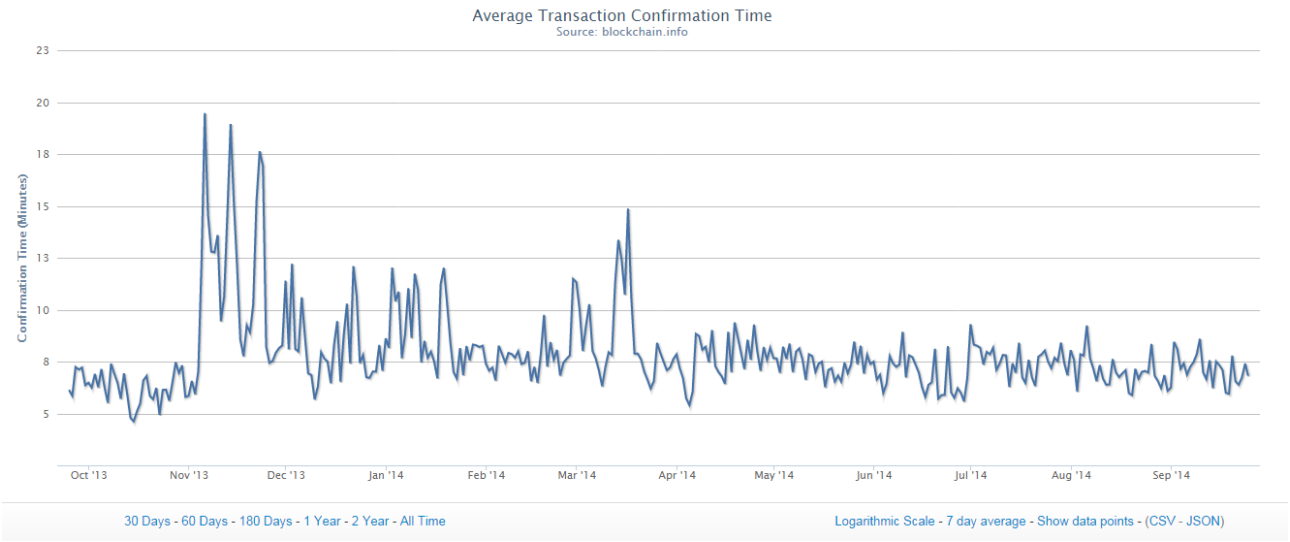

Рисунок 1.5 – Графік середнього часу підтвердження транзакцій

Цей графік відображає середній час в хвилинах, який йде на включення транзакції в блок. Тимчасові показники можуть коливатися в розумних межах, в

залежності від часу, які потрібні для валідації транзакції, а ступінь ризику залежить від обсягу транзакції [3].

1.1.10 Загальний обсяг блокчейну

Загальний обсяг блокчейну важливий з тієї причини, що для ефективної роботи мережі важливо, щоб обсяг місця на диску для зберігання інформації не перевищував розумних меж. І до того ж, розмір блокчейну впливає на швидкість синхронізації після встановлення нової версії клієнта. Цей метод вимірювання допомагає визначити істинні розміри блокчейну, включаючи хедери і транзакції, але виключаючи індексну базу даних

1.1.11 Стандартний розмір блоків

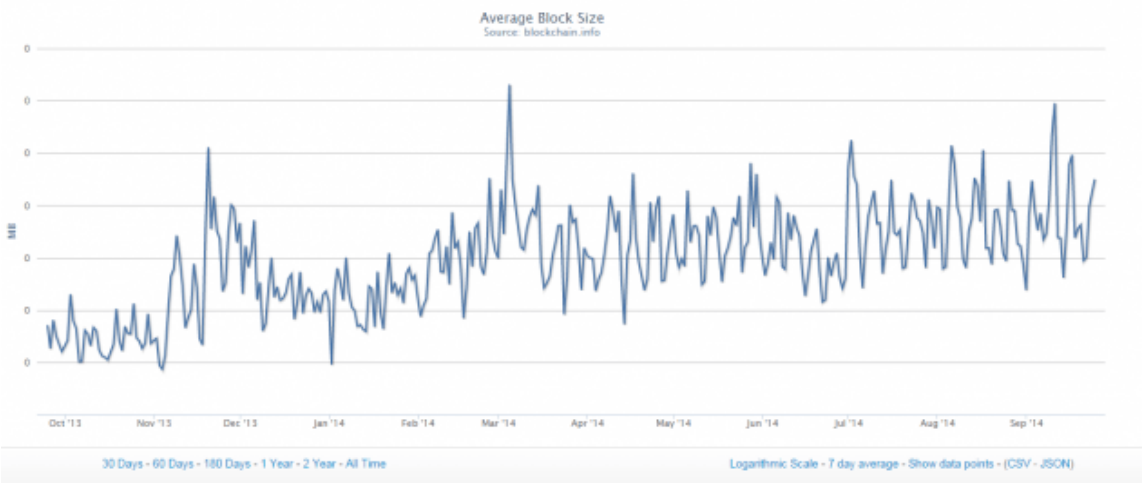

Рисунок 1.6 – Графік зміни розміру блоку у часі

Поки що стандартний розмір блоків вимірюється в частках мегабайта, але близький той час, коли їх розмір стане предметом палких суперечок - коли мережа наблизиться до граничним показниками продуктивної потужності, здійснюючи до 7 транзакцій в секунду.

Деякі вважають, що для планомірного розширення мережі вкрай важливо, щоб стандартний розмір блоків був збільшений, інші наполягають, що тут слід почекати,

щоб почали діяти ринкові механізми по визначенню реальної вартості транзакцій. Кожен з цих варіантів буде мати певні наслідки для майбутнього децентралізованої мережі.

1.2 Кластеризація адрес блокчейну

Розглянемо мережі на основі блоків, які допомагають об’єднати групи адрес блокчейн в одну суцільну систему. Ці показники засновані на певних моделях, які є загальними для багатьох транзакцій в мережі [4]. Однак вони не завжди задовольняються для всіх транзакцій, а отже схильні до помилок. Це означає, що деякі адреси можуть бути помилково пов'язані між собою.

Для аналізу транзакцій, окрема транзакція розглядається як упорядкована $t = (A, B, c)$ [5] та складається з:

- кінцевого багатоступеневого транзакційного входу A, де кожен вхід $(a_i, A_i) \in A$ – упорядкована пара адреси A_i і значення вхідного $a_i > 0$.
- кінцевого багатоступеневого транзакційного виходу B, де кожен вихід $(b_j, B_j) \in B$ – це упорядкована пара адреси B_j і значення вихідного $b_j \geq 0$.
- плати за транзакцію c $= \sum_{(a_i,) \in A} a_i - \sum_{(b_j,) \in B} b_j \geq 0$

Для довільної множини транзакційних входів або виходів A позначаємо мультимережний адрес в A, як Addr(A).

Найбільш очевидною ідеєю для кластеризації адрес блокчейну є з'єднання всіх вхідних адрес однієї транзакції. Якщо дві або більше адрес є входами однієї транзакції з одним виходом, то всі ці адреси керуються тим самим користувачем.

Розглянемо транзакцію t = (A, B, c), що задовольняє умови одноразової зміни.

а) #Addr (B) = 2, тобто транзакція t має рівно два виходи;

б) #Addr(A) = 6 ≠ 2, тобто кількість входів t не є рівною двом. Якщо #Addr(A) = #Addr(B) = 2, транзакція, швидше за все, поділиться міткою передачі;

в) обидва виходи транзакції t, B_1 та B_2 не є обмінними адресами, тобто $B_1, B_2 \notin \text{Addr}(A)$;

г) один вихід транзакції B_1 не існував до транзакції t, а десяткове подання значення b_1 має більше ніж 4 цифри після крапки;

д) інший вихід транзакції B_2 раніше був частиною мережі, і в попередніх транзакціях він не був адресований поза обліковим записом.

Розглянемо алгоритм кластеризації адрес на прикладі мережі Bitcoin, який регулює баланс інформації, що надходить безпосередньо з блоків Bitcoin (CS та OTC) та додаткову інформацію, зібрану з Інтернету у вигляді тегів.

Нехай $T = \{t_j\}$ і є набором всіх транзакцій в блоці біткойн, тоді як A є набором всіх адрес, що присутні в транзакції з T.

Кластеризація адрес Bitcoin – це розбивка $A = A_1 \cup A_2 \cup ... \cup A_N$ на непересічні підмножини $A_l \cap A_l = \emptyset$ для $l \neq$ j. За допомогою $T_H \subset$ T позначимо сукупність всіх транзакцій, які задовольняють CS, або OTC. Для транзакції $t \in T_H$, через $\text{Addr}_H(t)$ позначимо множину всіх адрес, які слід віднести до одного користувача відповідно.

Інформація про теги представлена як сукупність негативних пар L = $\{(a_i, a_j,)\}$. Пара адрес $(a_i, a_j) \in$ L, якщо у нас є частина інформації про те, що ці адреси не контролюються одним і тим самим користувачем.

Слід зазначити, що як CS і OTC, так і позабіржова евристика і набір негативних пар L можуть містити помилкову інформацію.

Розглянемо різні типи спостережень:

– у випадку, якщо всі адреси $\text{Addr}_H(t)$ для деяких t $\in T_H$ дійсно належать одному і тому ж користувачу з ймовірністю p;

– у випадку, якщо дві адреси $(a_i, a_j) \in L$ контролюються тим самим користувачем з ймовірністю q.

В інших випадках інформація про негативне об'єднання між будь-якою парою адрес в L перевіряється шляхом 1 - q.

Нехай ймовірність $P(A, T_H, L|p, q)$ буде функцією від кластеризації A, транзакції T_H та негативних пар L:

$$P(A, T_H, L|p, q) = \\ = \prod_{\mathrm{t} \in T_H} p^{\mathbb{I}(\mathrm{Addr}_H\,(\mathrm{t}) \subset Cl(A))} \times (1-p)^{\mathbb{I}\left(\mathrm{Addr}_H\,(\mathrm{t}) \not\subset Cl(A)\right)} \times \\ \times \prod_{\{a,a\prime\} \in \mathrm{L}} (1-q)^{\mathbb{I}(\{a,a\prime\} \not\subset Cl(A))} \times q^{\mathbb{I}(\{a,a\prime\} \subset Cl(A))}, \tag{1.1}$$

де p – ймовірність того, що адреси $\mathrm{Addr}_H(t)$ належать одному користувачу, q – ймовірність того, що пара адрес $(a_i, a_j) \in L$ належать одному користувачу, для деякого набору Bitcoin адреси S позначення $S \subset Cl\,(A)$ означає, що існує кластер A_l, такий, що $S \subseteq A_l$, та $\mathbb{I}(x) = 1$, якщо твердження x вірне, та $\mathbb{I}(x) = 0$, якщо – ні.

Отже, log-правдоподібність співвідноситься як

$$\ln P(A, T_H, L|p, q) = \sum_{T \in T_H} \mathbb{I}\left(\mathrm{Addr}_H\,(\mathrm{t}) \subset Cl(A)\right) \ln(1-p) + \\ + \sum_{T \in T_H} \mathbb{I}\left(\mathrm{Addr}_H\,(\mathrm{t}) \not\subset Cl(A)\right) \ln(p) + \\ + \sum_{\{a,a\prime\} \in \mathrm{L}} \mathbb{I}\left(\{a, a'\} \not\subset Cl(A)\right) \ln(1-p) + \\ + \sum_{\{a,a\prime\} \in \mathrm{L}} \mathbb{I}(\{a, a'\} \subset Cl(A)) \ln(p). \tag{1.2}$$

Максимізація log-правдоподібності – це задача дискретної оптимізації, яка фактично NP-повна.

В [5] наведено евристику, які задовільняють транзакції в мережі Bitcoin. На кожному етапі вирішується, чи приєднуються кластери, що відповідають адресі $\mathrm{Addr}_H\,(t_J)$ до розглянутої транзакції t_J.

Нехай $\hat{A}_j = A_{k_1} \cup \ldots \cup A_{k_m}$ – об'єднання всіх кластерів, представники яких належать $\mathrm{Addr}_H\,(t_J)$.

Знайдемо зміни кількості негативних пар, що відповідають $\mathrm{Addr}_H\ (t_J)$ в один кластер A_J:

$$\Delta_{t_j}\left(\sum_{\{a,a'\}\in \mathrm{L}} \mathbb{I}\big(\{a,a'\}\not\subset Cl(A)\big)\right)=\sum_{\{a,a'\}\in \hat{A}_J} \mathbb{I}(\{a,a'\}\in A_l)- \\ -\sum_{i=1}^{m_j}\sum_{\{a,a'\}\in A_{k_i}} \mathbb{I}\big(\{a_i,a_j,\}\in A_l\big)=\Delta_{\hat{A}_j}-\sum_{i=1}^{m_j}\Delta_{A_j}, \tag{1.3}$$

де ΔA_m – це кількість негативних пар в кластері A_m, $\hat{A}_j$ – об'єднання всіх кластерів, представники яких належать $\mathrm{Addr}_H\ (t_J)$.

Тепер об'єднаємо всі кластери, що відповідають $\mathrm{Addr}_H\ (t_J)$, а отже зміна log-правдоподібності дорівнює

$$\Delta_P\big(t_j,A,L\big|p,q\big)=ln\left(\frac{p}{1-p}\right)+\left(\Delta_{\hat{A}_j}-\sum_{i=1}^{m_j}\Delta_{A_j}\right)\ln\left(\frac{q}{1-q}\right). \tag{1.4}$$

Таким чином, якщо $\Delta_P\big(t_j,A,L\big|p,q\big)$ є позитивним, то ми зливаємо всі кластери, що відповідають $\mathrm{Addr}_H\ (t_J)$, в іншому випадку потрібно продовжувати наступну транзакцію.

1.3 Огляд платформи BlockSci

Аналіз даних блокчейн корисний для наукових досліджень і комерційних програм. Суттєве місце в аналізі займає BlockSci – відкрита програмна платформа для аналізу блочного діапазону. BlockSci є універсальним для підтримки різних блоків і аналізу завдань. Він включає в себе аналітичну базу даних в пам'яті, що робить його в кілька сотень разів швидше, ніж існуючі інструменти [6].

Розглянемо архітектуру BlockSci і методи аналізу, які ілюструють його можливості.

Спочатку необхідно визначити вхід від мережі для подальшого розвитку програмного забезпечення та вивчення інших потенційних застосувань.

Існує два способи імпорту даних у BlockSci. Через будь-який маршрут, дані перетворюються на той ж самий проміжний формат для аналізу. Розроблено синтаксичний аналізатор даних основного блочного кешу, який поступово може бути оновлено, адже з'являються нові блоки. Бібліотека аналізу завантажує ці дані як базу даних в пам'яті, яку користувач може запитати через інтерфейс ноутбука.

Блокчейн складається в основному з спрямованого ациклічного графу транзакцій. У транзакціях, що з'єднуються ребрами, є атрибути, тобто адреси або скрипти, прикріплені до них. Транзакції згруповані в блоки, розташовані в лінійній ланцюжку, з невеликою кількістю метаданих на блок. BlockSci підтримує блоки, які дотримуються цієї базової структури.

Наприклад, Litecoin не вносить жодних змін у структуру даних, і, таким чином, повністю підтримується. Криптовалюти, які вносять зміни до операцій скрипту, можуть підтримуватися лише частково. Namecoin підтримується, але новий тип скрипту, який він представляє, не аналізується BlockSci (користувач може проаналізувати їх декількома рядками коду). Zcash також підтримується, принаймні, в тому сенсі, що аналіз блокчейн Zcash навіть можливий: він представляє собою складний сценарій, що включає докази нульового знання, але ці аспекти відокремлюються в спеціальному виді адреси, який не є загальнодоступним за дизайном.

Прикладом поточного непідтримуваного блочного шаблону є Monero, оскільки він не відповідає парадигмі "один вхід, один вихід".

BlockSci також записує дані Mempool, тобто інформацію про транзакції, які транслюються в мережу P2P і чекають, що вони будуть включені до блоку. Час очікування транзакцій забезпечує цінні дані про ринок блочного простору (і не фіксується в блочному каналі) [7].

Реєстратор Mempool має два режими. У мінімальному режимі він записує лише тимчасові позначки (еквівалентно часу очікування) транзакцій, які перетворили його

в блок-схему. Слід зауважити, що загальнодоступні джерела даних mempool дозволяють запитувати мітку часу за допомогою хеш-транзакції, але не дозволяють масово завантажувати ці дані.

У повному режимі рекордер включає в себе всю інформацію в mempool, яка охоплює транзакції, які ще не були включені в блок. Дані часової проміжки завантажуються в пам'ять для аналізу, тоді як дані повного режиму зберігаються на диску.

У будь-якій системі однорангової мережі різні вузли отримають однакові дані в різний час.

BlockSci - це система з одним вузлом, тому її часові мітки неминуче відстають. Будь-який користувач BlockSci може виконувати необхідні вимірювання проміжків часу та застосувати рівномірну корекцію для усунення середнього відставання, але, звичайно, дисперсія залишиться.

Формат блоків на диску є неефективним. Він оптимізований для різних цілей, таких як перевірка транзакції та забезпечення незмінності. Існує ряд способів, що дозволяють мінімізувати споживання пам'яті за рахунок простору диска та одночасно оптимізувати швидкість доступу:

- вивести посилання на входи, які витрачають їх, щоб дозволяти перетинати графік;
- замінити хеш-покажчики з ідентифікаторами, щоб зменшити структуру даних і оптимізувати зв'язок;
- використовувати кодировки розміру для даних, коли це можливо;
- дублювати дані адреси/скрипта;
- оптимізувати макет пам'яті для місцевості посилання.

Система блокчейн повинна оброблятися послідовно, щоб перетворити блок у формат аналізу BlockSci. Кожна специфікація вхідної транзакції, закодована як і вихідна. Щоб перетворити хеш транзакції в ідентифікатор, аналізатор повинен зберегти хеш-мапу ідентифікаторів. Аналогічним чином, він повинен підтримувати відображення адрес ідентифікаторів для дедуплікації.

Перетворення хешу транзакції у ідентифікатор може бути зменшене шляхом обрізки хеш-транзакцій, для яких всі виходи транзакції було витрачено. Однак, відображення адрес не дає такої оптимізації. Будь-яка адреса може бути використана будь-яким виходом. Таким чином, усі адреси треба відстежувати. Зберігання карти в пам'яті і на диску зробить аналізатор дуже повільним.

Щоб досягти подальшої оптимізації, слід відзначити, що переважна більшість ресурсів витрачають нещодавно створені результати (наприклад, 89% витрат ідуть на створення останніх 4000 блоків). Точно так само переважна більшість адрес, використовуються після їх початкового використання знову (наприклад, 90% у межах 4000 блоків).

BlockSci в даний час ділить скрипти на 5 типів: pay-to-public-key-hash, pay-to-script-hash, multisig, pubkey і нульові дані (OP_RETURN). Усі інші скрипти класифікуються як нестандартні.

Для скриптів, що відносяться до будь-якого з підтримуваних типів, BlockSci аналізує сценарій та зберігає інформацію, що стосується аналізу, при відмові від непотрібних даних сценарію.

Для pubkey і pay-to-public-key-hash це означає, що записується pubkeyhash та pubkey, коли це доступно (наприклад, якщо вихід був витрачений). Для pay-to-script-хешу записується хеш-сценарій, а також посилання на його адресу. Для multisig реєструються вказівки на адреси, які можуть витрачати multisig, а також кількість адрес, необхідних для їх витрачання. Для нульових даних записується відповідний скрипт. Для нестандартних типів записується весь сценарій, дозволяючи користувачеві, коли це необхідно, написати власний синтаксичний аналіз.

Аналізуючи мережу блокчейн, необхідно розглянути картографію пам'яті та паралелізм. Оскільки BlockSci використовує той самий формат для графіка транзакцій на диску та в пам'яті, завантаження блокчейну включає в себе відображення пам'яті. Після того, як в пам'яті по кожній транзакції отримали C++ struct, жодна нова пам'ять не повинна бути виділена, щоб включитися до орієнтованого інтерфейсу даних.

Наступні три властивості притаманні BlockSci:

- таблиця транзакцій постійно оновлюється на диску після отримання нових блоків (старі операції можуть бути оновлені, якщо вони мають невитрачені виходи, які витрачаються на нові блоки);
- таблиця на карті пам'яті і розподілені між усіма запущеними екземплярами BlockSci;
- кожен екземпляр завантажує знімок блокчейну, який ніколи не змінюється, якщо програміст не викликає перезавантаження.

Стан таблиці операцій у будь-який минулий момент часу (висота блоку) може бути реконструйований з урахуванням поточного стану.

Щоб забезпечити ілюзію статичної структури даних, коли об’єкт блокчейн ініціалізується, атрибут maxHeight зберігає висоту блочного шару під час ініціалізації. Висота блокчейн на диску з часом зростає, але атрибут maxHeight залишається, адже має незмінний показник. Тому доступ до блоків, що перевищують цю висоту, неможливий. Бібліотека аналізу перехоплює доступ до транзакційних виходів і перезаписує їх таким чином, що виходи, які витрачаються на блоки після maxHeight, трактуються як невитрачені.

BlockSci в даний час відкриває тільки найдовші ланцюги і ховає застарілі блоки. Інакше кажучи, вона спрямована на приховування реорганізацій блокчейн. Це робиться, ігноруючи останні кілька блоків під час ініціалізації. Імовірність повторення того, що кількість елементів d або більше блоків зменшується експоненціально в d. Значення за замовчуванням d становить 6. Якщо відбувається глибша реоргація, бібліотека аналізу видає виняток.

Багато завдань аналізу, таких як обчислення середньої вартості транзакції з часом, можна виразити як зменшення кількості операцій над таблицею транзакцій (або діапазонів блоків). Таким чином, бібліотека аналізу підтримує карту зменшення абстракції. Додатковою перевагою є паралелізм: користувач бібліотеки обробляє завдання, щоб використовувати всі наявні ядра.

Користувачі криптовалюти можуть тривіально генерувати нові адреси, і більшість з них користуються перевагами цієї можливості. Тим не менше, адреси, керовані одним і тим самим користувачем або юридичною особою, можуть бути пов'язані одна з одною, хоча і недосконало, через різні евристики. Зв'язок з адресою є ключовим кроком у аналітичних завданнях, включаючи розуміння тенденцій у часі та оцінки приватності.

В [6] запропоновано дві евристики для аналізу адрес:

- входи, витрачені на одну транзакцію, контролюються одним і тим самим об'єктом;
- адреси змін не використовуються повторно.

Але є і виключення: це не стосується транзакцій CoinJoin.

Ці евристики створюють посилання (ребра) в графі адрес. За допомогою повторення всіх транзакцій та застосування алгоритму Union на графі адрес, ми можемо генерувати кластери адрес.

Цей набір кластерів - це вихідний зв'язок адреси. Зв'язування адреси за своїм характером є недосконалим, і важко досягти повної відповідності у великих масштабах, оскільки це вимагає взаємодії з постачальниками послуг. Можливі багато інших евристик, в тому числі ті, що пояснюють поведінку конкретних рахунків.

Більшість евристик піддаються помилковим негативам, що може призвести до "колапсу кластеру". Припустимо, що спектральні методи кластеризації можуть мінімізувати помилкові позитиви та негативи і в значній мірі усувають потребу у складеній ручній евристиці. Посилання на адреси є особливо потужними, коли поєднуються з маркуванням адрес, тобто адресними позначками з ідентифікаціями в реальному світі. Це може бути корисним для криміналістики та правоохоронних органів, але це також може порушити конфіденційність користувачів.

Такі компанії, як Chainalysis та Elliptic, спеціалізуються на тегах і криміналістиці. BlockSci має обмежену функцію тегів: якщо користувач надає теги для підмножини адрес, алгоритм зв'язання адреси поширюватиме ці теги під час етапу посилання на адресу.

Свідомі користувачі або компанії, які зберігають велику кількість криптовалюти, часто використовують багатовекторність можливостей блокчейн. На відміну від стандартних транзакцій з оплатою за відкритим ключем-хешем (P2PKH), для підписування яких потрібна лише один підпис, багатозв'язні адреси дозволяють вказувати n ключів та параметр $m \leq n$, для яких потрібно ввести m певних ключів, щоб витратити гроші.

Ця функція дозволяє розподілити контроль над блокчейн-гаманцем: ключі можуть зберігатися на n серверах або в n різних співробітників компанії, так, щоб m з них повинні були погоджуватися на авторизацію транзакції.

Типовим прикладом цього є те, що користувач може зберігати ключі на своєму настільному комп'ютері і на смартфоні, і вимагати авторизації на обох пристроях.

Висновки до розділу

При проведенні аналізу літературних джерел були оцінені основні сучасні методи оцінки та метрики для ланцюгів транзакцій блокчейну і виявлені їх недоліки, основними з яких є:

- визначення нових характеристик без надання певних висновків;
- використовується тільки одна характеристика системи для визначення метрик;
- аналіз поведінки користувачів тільки з точки зору роботи системи.

Зазначені недоліки є досить вагомими і роблять практично неможливим ефективне застосування даних методів в потужних сервісах з аналізу та моніторингу блокчейн.

Метою роботи є підвищення ефективності використання інструментів аналізу транзакцій, шляхом впровадження елементів кластерного аналізу даних, що потребує розроблення комплексного підходу до розв‘язання задачі кластеризації адресів блокчейн.

Основні завдання, які мають бути виконані для досягнення цієї мети:

а) визначити необхідні параметри для вирішення задачі кластеризації;

б) розв‘язати задачу кластеризації адресів корисутвачів в мережі блокчейну;

в) виконати програмну реалізацію алгоритмічного забезпечення.

2 ОГЛЯД МЕТОДІВ КЛАСТЕРИЗАЦІЇ

2.1 Зміст задачі кластеризації

Кластерний аналіз — задача розбиття заданої вибірки об'єктів (ситуацій) на підмножини, що називаються кластерами, так, щоб кожен кластер складався з схожих об'єктів, а об'єкти різних кластерів істотно відрізнялися [8]. Завдання кластеризації відноситься до статистичної обробки, а також до широкого класу завдань навчання без вчителя. Основна мета кластерного аналізу — знаходження груп схожих об'єктів у вибірці. Спектр застосувань кластерного аналізу дуже широкий: його використовують в археології, антропології, медицині, психології, хімії, біології, державному управлінні, філології, маркетингу, соціології та інших дисциплінах.

Кластерний аналіз — це багатовимірна статистична процедура, яка виконує збір даних, що містять інформацію про вибірку об'єктів і потім упорядковує об'єкти в порівняно однорідні групи — кластери. Кластеризація є описовою процедурою, вона не робить ніяких статистичних висновків, але дає можливість провести розвідувальний аналіз і вивчити "структуру даних". Саме поняття "кластер" визначено неоднозначно: в кожному дослідженні свої "кластери". Перекладається поняття кластер як "скупчення", "гроно". Кластер можна охарактеризувати як групу об'єктів, що мають загальні властивості. Слід зазначити, що в результаті застосування різних методів кластерного аналізу можуть бути отримані кластери різної форми. Наприклад, можливі кластери "ланцюгового" типу, коли кластери представлені довгими "ланцюжками", кластери подовженої форми і т.д., а деякі методи можуть створювати кластери довільної форми.

Різні методи можуть прагнути створювати кластери певних розмірів (наприклад, малих або великих) або припускати в наборі даних наявність кластерів різного розміру. Деякі методи кластерного аналізу особливо чутливі до шумів або викидам, інші - менш. В результаті застосування різних методів кластеризації можуть бути отримані неоднакові результати, це нормально і є особливістю роботи того чи іншого алгоритму. [10]

2.2 Завдання кластеризації

Кластерний аналіз виконує наступні основні завдання:

- розробка типології або класифікації;
- дослідження корисних концептуальних схем групування об'єктів;
- породження гіпотез на основі дослідження даних;
- перевірка гіпотез або дослідження для визначення, чи дійсно групи, виділені тим чи іншим способом, присутні в наявних даних [9].

Як правило, кластеризація застосовується для того, щоб здійснити стиснення даних для скорочення обсягу використовуваних даних за рахунок того, що всередині кластера об'єкти не розрізняються (розглядаються як один об'єкт).

Кластеризація може забезпечити краще розуміння даних і спростити їх подальшу обробку. Наприклад, сегментування ринку за будь-якими ознаками поведінки споживача дозволяє проводити адресні акції та різні маркетингові заходи, спрямовані на збільшення обсягу продажів.

При проведенні кластеризації можуть виявитися нетипові об'єкти, які не можна віднести ні до одного з класів - це може дати новий корисний матеріал для дослідження.

2.3 Загальна математична постановка задачі кластеризації

Задача кластеризації (або навчання без вчителя) полягає в наступному. Мається навчальна вибірка $X^j = \{x_1 \dots x_i\} \subset X$ і функція відстані між об'єктами $p = (x, x')$ Потрібно розбити вибірку на непересічні підмножини, які називаються кластерами, так, щоб кожен кластер складався з об'єктів, близьких по метриці p, а об'єкти різних кластерів істотно відрізнялися. При цьому кожному об'єкту $x_i \in X^j$ приписується мітка (номер) кластера y_i [11].

Алгоритм кластеризації - це функція $a: X \to Y$, яка будь-якому об'єкту $x \in X$ ставить у відповідність мітку кластера $y \in Y$. Множину міток Y в деяких випадках відомо заздалегідь, однак частіше ставиться завдання визначити оптимальне число

кластерів, з точки зору того чи іншого критерію якості кластеризації. Рішення завдання кластеризації принципово неоднозначно, і тому є кілька причин:

- не існує однозначно найкращого критерію якості кластеризації. Відомий цілий ряд досить розумних критеріїв, а також ряд алгоритмів, які не мають чітко вираженого критерію, але здійснюють достатньо розумну кластеризацію «з побудови». Всі вони можуть давати різні результати;
- число кластерів, як правило, невідомо заздалегідь і встановлюється відповідно до деякого суб'єктивного критерію;
- результат кластеризації істотно залежить від метрики ρ, вибір якої, як правило, також суб'єктивний і визначається експертом.

2.4 Підходи до вирішення задачі кластеризації

На рисунку 2.1 представлена класифікація алгоритмів та методів кластерного аналізу [11].

Сутність ієрархічних агломеративних методів полягає у тому, що на першому кроці кожний об’єкт вибірки розглядається як окремий кластер. Процес об’єднання кластерів відбувається послідовно, на підставі матриці відстаней або матриці подібності поєднуються найбільш близькі об’єкти. Послідовність об’єднання легко піддається геометричній інтерпретації й може бути представлена у вигляді графа-дерева. Основною передумовою ієрархічних дивізійних методів є те, що спочатку всі об’єкти належать до одного кластеру. У процесі класифікації за певними правилами поступово від цього кластера відокремлюються групи схожих між собою об’єктів. Так, на кожному кроці кількість кластерів зростає, а міра відстані між кластерами зменшується. Складнощі ієрархічних методів кластеризації наступні:

- обмеження обсягу набору даних;
- вибір міри близькості;
- негнучкість отриманих класифікацій.

Перевага цієї групи методів порівняно з неієрархічними методами полягає у їх наочності і можливості отримання детального уявлення про структуру даних. При використанні ієрархічних методів існує можливість досить легко ідентифікувати викиди в наборі даних і в результаті підвищити якість даних. Велика кількість методів ієрархічного кластерного аналізу різниться не тільки використаними мірами подібності (розходження),але й алгоритмами класифікації.

Неієрархічні методи виявляють більш високу стійкість по відношенню до викидів, невірного вибору метрики, включення незначущих змінних в базу для кластеризації та інше. Необхідно заздалегідь фіксувати результуючу кількість кластерів, правило зупинки і, якщо на те є підстави, початковий центр кластеру, що суттєво впливає на ефективність роботи алгоритму. Якщо немає підстав штучно задавати ці умови, рекомендується використовувати ієрархічні методи [11].

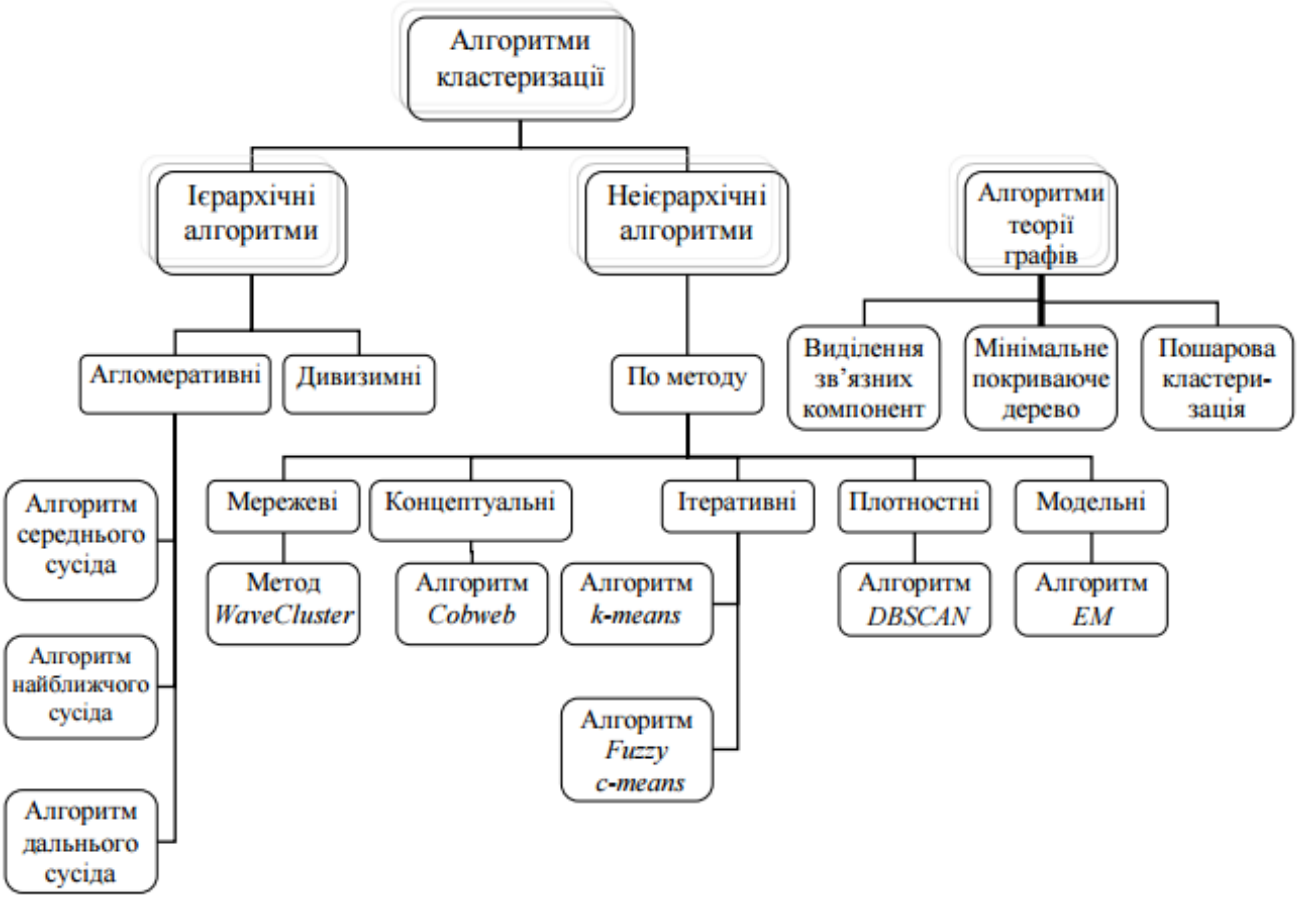

Рисунок 2.1 – Класифікація алгоритмів та методів кластерного аналізу

2.5 Алгоритм ієрархічної кластеризації

Серед алгоритмів ієрархічної кластеризації виділяються два основні типи: агломеративні і дивизимні алгоритми. Дивизимні алгоритми працюють за принципом «зверху-униз»: на початку всі об'єкти поміщаються в один кластер, який потім розбивається на більш дрібні кластери. Більш поширені агломеративні алгоритми, які на початку роботи поміщають кожний об'єкт в окремий кластер, а потім поєднують кластери в більш великі, поки всі об'єкти вибірки не будуть утримуватися в одному кластері. У такий спосіб будується система вкладених розбивок. Результати таких алгоритмів звичайно представляють у вигляді дендрограми. Класичним прикладом такого дерева є класифікація тварин і рослин. До недоліку ієрархічних алгоритмів можна віднести систему повних розбивок, яка може бути зайвою в контексті розв'язуваної задачі [11].

2.6 Алгоритми квадратичної помилки

Задачу кластеризації можна розглядати як побудову оптимальної розбивки об'єктів на групи. При цьому оптимальність може бути визначена як вимога мінімізації середньоквадратичної помилки розбивки:

$$e^2(X, L) = \sum_{j=1}^{K} \sum_{i=1}^{n_j} \left\| x_i^{(j)} - c_j \right\|, \qquad (2.1)$$

де c_j – «центр мас» кластеру j (точка із середніми значеннями характеристик для даного кластеру).

Найпоширенішим алгоритмом цієї категорії є метод k-середніх. Цей алгоритм будує задане число кластерів, розташованих максимально віддалено один від одного. Робота алгоритму ділиться на кілька етапів:

а) випадкове обрання k точок, які є початковими «центрами мас» кластерів;

б) віднесення кожного об'єкту до кластеру з найближчим «центром мас»;

в) перерахунок «центрів мас» кластерів згідно з їхнім поточним складом;

г) повернення до п.2 за умови, що критерій зупинки алгоритму не виконано.

У якості критерію зупинки роботи алгоритму звичайно вибирають мінімальну змінну середньоквадратичної помилки. Також можливо припиняти роботу алгоритму за умови, якщо на кроці 2 не було об'єктів, які перемістилися із кластера в кластер. Слід зазначити, що основним недоліком алгоритмів на базі методу k-середніх є вимога початкового визначення кількості та положення центрів кластерів. Інформація про ці параметри на початковому етапі дослідження інформаційного простору, як правило, відсутня.

2.7 Ієрархічні методи

Сутність ієрархічних агломеративних методів у тому, що на першому кроці кожний об’єкт вибірки розглядається як окремий кластер. Процес об’єднання кластерів відбувається послідовно: на підставі матриці відстаней або матриці подібності поєднуються найбільш близькі об’єкти. Послідовність об’єднання легко піддається геометричній інтерпретації й може бути представлена у вигляді графа-дерева (дендрограми).

Основною передумовою ієрархічних дивізімних методів є те, що спочатку всі об’єкти належать до одного кластеру. У процесі класифікації за певними правилами поступово від цього кластера відділяються групи схожих між собою об’єктів. Таким чином, на кожному кроці кількість кластерів зростає, а міра відстані між кластерами зменшується.

При використанні ієрархічних методів існує можливість досить легко ідентифікувати викиди в наборі даних і, в результаті, підвищити якість даних.

Велика кількість методів ієрархічного кластерного аналізу різниться не тільки використовуваними мірами подібності (розходження), але й методами об’єднання об’єктів у кластери.

Розглянемо принцип роботи групи ієрархічних методів (рисунок 2.2).

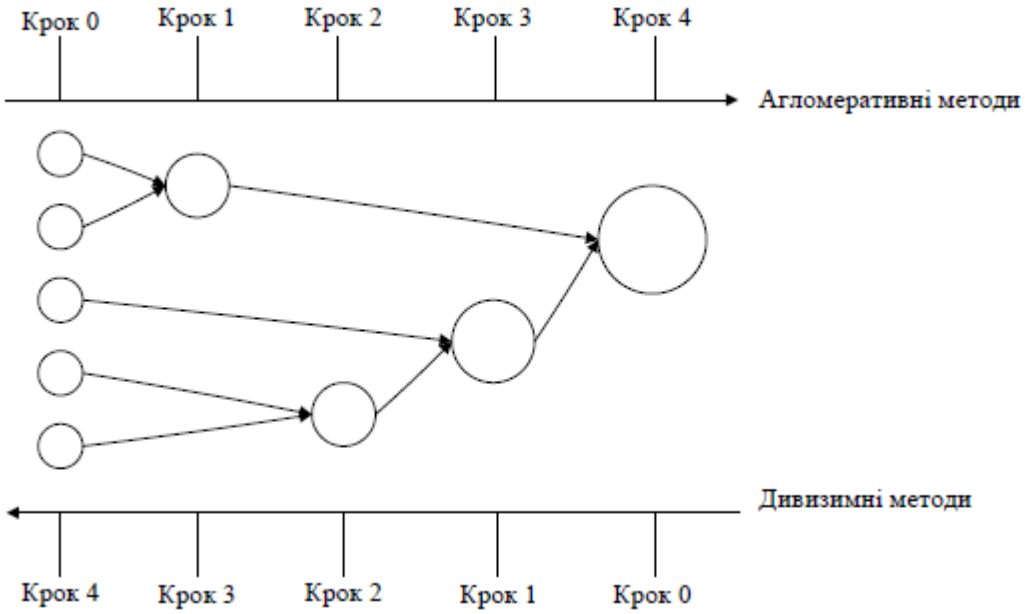

Рисунок 2.2 – Принцип роботи ієрархічних методів

2.7.1 Правила об’єднання кластерів

На першому кроці, коли кожний об’єкт являє собою окремий кластер, відстані між цими об’єктами визначаються обраною мірою. Однак коли зв’язуються разом кілька об’єктів, виникає питання, як слід визначити відстані між кластерами. Іншими словами, необхідно правило об’єднання або зв’язку для двох кластерів.

Існує декілька методів об’єднання кластерів:

– одиночний зв’язок (метод найближчого сусіда). У цьому методі відстань між двома кластерами визначається відстанню між двома найбільш близькими об’єктами (найближчими сусідами) у різних кластерах (рисунок 2.3).

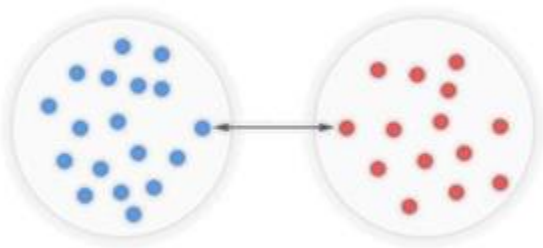

Рисунок 2.3 – Метод найближчого сусіда

Переваги методу:

- простота використання отриманих результатів;
- рішення не унікальні для конкретної ситуації, можливо їх використання для інших випадків;
- метою пошуку є не гарантовано вірне рішення, а краще з можливих.

Недоліки методу:

- даний метод не створює будь-яких моделей або правил, узагальнюючих попередній досвід, у виборі рішення вони ґрунтуються на всьому масиві доступних історичних даних, тому неможливо сказати, на якій підставі будуються відповіді;
- складність вибору міри «близькості» (метрики). Від цієї міри головним чином залежить обсяг множини записів, які потрібно зберігати в пам'яті для досягнення задовільної класифікації або прогнозу. Також існує висока залежність результатів класифікації від обраної метрики;
- при використанні методу виникає необхідність повного перебору навчальної вибірки при розпізнаванні, наслідок цього – обчислювальна трудомісткість;
- типові завдання даного методу – це завдання невеликої розмірності за кількістю класів і змінних.
- повний зв'язок (метод найбільш віддалених сусідів). У цьому методі відстані між кластерами визначаються найбільшою відстанню між будь-якими двома об'єктами в різних кластерах (рисунок 2.4).

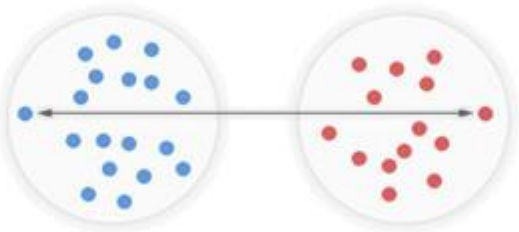

Рисунок 2.4 – Метод найвіддаленішого сусіда

Перевага методу – формуються відносно компактні гіперсферичні кластери, що складаються з об’єктів з великою схожістю.

Недолік методу – непридатність методу в разі подовжених форм кластерів.

– незважене попарне середнє. У цьому методі відстань між двома різними кластерами обчислюється як середня відстань між усіма парами об’єктів в них.

Перевага методу в тому, що коли об’єкти формують кластери різних розмірів, проте він працює однаково добре і у випадках протяжних («ланцюгового» типу) кластерів.

– зважене попарне середнє. Метод ідентичний до методу незваженого попарного середнього, за винятком того, що при обчисленнях розмір відповідних кластерів (тобто число об’єктів, що містяться в них) використовується в якості вагового коефіцієнта. Тому запропонований метод повинен бути використаний, коли передбачаються нерівні розміри кластерів. Методу ефективний у разі коли передбачаються нерівні розміри кластерів.
– незважений центроїдний метод (рисунок 2.5). У цьому методі відстань між двома кластерами визначається як відстань між їхніми центроїдами.

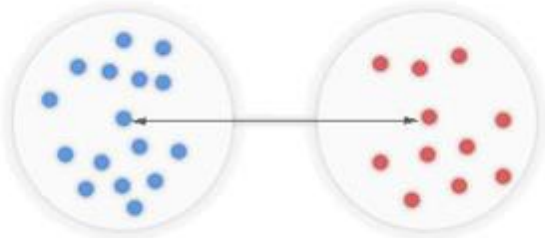

Рисунок 2.5 – Незважений центроїд

Недолік методу – у разі об’єднання двох кластерів, в одному з яких багато елементів, а в іншому – мало, характеристики останнього практично ігноруються.

– зважений центроїдний метод (медіана). Цей метод ідентичний попередньому, за винятком того, що при обчисленнях використовуються

ваги для обліку різниці між розмірами кластерів (тобто числами об’єктів в них). Тому, якщо є (або підозрюються) значні відмінності в розмірах кластерів, варто віддати перевагу цьому методу. Метод ефективний, якщо є значні відмінності в розмірах кластерів.

– метод Варда (рисунок 2.6). Цей метод відрізняється від усіх інших методів, оскільки він використовує методи дисперсійного аналізу для оцінки відстаней між кластерами. Метод мінімізує суму квадратів для будь-яких двох кластерів, які можуть бути сформовані на кожному кроці.

При використанні методу Варда дендрограмма виходить з глибоко розділеними, більш «виразними» та найбільш однорідними в статистичному сенсі кластерами. У цілому метод видається дуже ефективним. При використанні цього методу створюються кластери приблизно рівних розмірів і які мають форму гіперсфер, що відповідає критерію якості кластеризації.

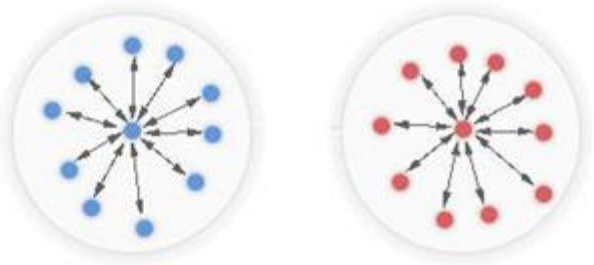

Рисунок 2.6 – Метод Варда

2.8 Алгоритм К-середніх

Найбільш поширений серед неієрархічних методів – алгоритм k-середніх. Для використання цього методу необхідно мати заздалегідь відому кількість кластерів.

Алгоритм k-середніх будує k кластерів, розташованих на можливо великих відстанях один від одного. Загальна ідея алгоритму: задане фіксоване число k кластерів спостереження зіставляються кластерам так, що середні в кластері (для всіх змінних) максимально можливо відрізняються один від одного.

Нехай є n спостережень, кожне з яких характеризується p ознаками $x_1, x_2 \dots x_n$. Ці спостереження необхідно розбити на k кластерів:

- з n точок досліджуваної сукупності відбираються випадковим чином або задаються дослідником виходячи з яких-небудь апріорних міркувань k точок (об'єктів)(ці точки приймаються за еталони);
- кожному еталону привласнюється порядковий номер, що є одночасно номером кластера;
- із n-k об'єктів, що залишилися, вибирається точка x_i з координатами $(x_{i2}, \dots, x_{ip})$ і перевіряється, до якого з еталонів (центрів) вона знаходиться ближче всього. Для цього використовується одна з метрик;
- об'єкт, що перевіряється, приєднується до того центра (еталону), якому відповідає $d_{ij}, (i = 1 \dots k)$;
- еталон замінюється новим(перерахованим) з урахуванням приєднаної точки, і вага його (кількість об'єктів, що входять до даного кластера) збільшується на одиницю. Якщо зустрічаються дві або більше мінімальних відстаней, то i-й об'єкт приєднують до центра з найменшим порядковим номером;
- вибирається наступна точка x_i і для неї повторюються всі процедури.

Таким чином, через $(n-k)$ кроків всі точки (об'єкти) сукупності виявляться віднесеними до одного з k кластерів, але на цьому процес розбивки не закінчується. Для того, щоб домогтися стабільності розбивки по тому ж правилу, всі точки $x_1, x_2 \dots x_n$ знову приєднуються до отриманих кластерів, при цьому ваги продовжують накопичуватися. Нова розбивка порівнюється з попередньою. Якщо вони збігаються, то робота алгоритму завершується, інакше цикл повторюється. Остаточна розбивка має центри ваги, які не збігаються з еталонами, їх можна позначити $c_1, c_2 \dots c_k$. При цьому кожна точка $x_i, i = 1 \dots n$ буде відноситися до такого кластера (класу), для якого відстань мінімальна.

Переваги алгоритму k-середніх:

- простота використання;
- швидкість використання;
- зрозумілість і прозорість алгоритму.

Недоліки алгоритму k-середніх:

- результат класифікації сильно залежить від випадкових початкових позицій кластерних центрів;
- не гарантується досягнення глобального мінімуму сумарного квадратичного відхилення, а лише одного з локальних мінімумів;
- кількість кластерів повинна бути заздалегідь визначена дослідником.

2.9 EM алгоритм

ЕМ алгоритм використовується в математичній статистиці для знаходження оцінок максимальної схожості параметрів імовірних моделей, у випадку, коли модель залежить від деяких прихованих змінних. Кожна ітерація алгоритму складається з двох кроків. На Е-кроці (expectation) вираховується очікуване значення функції правдоподібності, при цьому приховані змінні розглядаються як спостережувані. На М-кроці (maximization) вираховується оцінка максимальної схожості, таким чином збільшується очікувана схожість, вирахувана на Е-кроку. Потім це значення використовується для Е-кроку на наступній ітерації. Алгоритм виконується до того часу, поки не буде виконана умова збіжності.

Загальна ідея алгоритму: нехай X — деяке з значень спостережуваних змінних, а T— прихованні змінні. Разом X і T утворюють повний набір даних. Взагалі, Tможе бути деякою підказкою, яка полегшує рішення проблеми у випадку, якщо вона відома. Наприклад, якщо є суміш розподілів, то функція правдоподібності легко виражається через параметри відокремлених розподілів суміші.

Нехай p – густина імовірності (в безперервному випадку) або функція імовірностей (в дискретному випадку) повного набору даних з параметрами Θ:$p(X,T|\Theta)$. Цю функцію можна розуміти як правдоподібність всієї моделі, якщо

розглядати її як функцію параметрів Θ. Зауважимо, що умовний розподіл прихованої компоненти при деякому спостереженні та фіксованому наборі параметрів може бути виражене так (використовуючи розширену формулу Байєса і формулу повної імовірності):

$$p(T|X,\Theta) = \frac{p(X,T|\Theta)}{p(X|\Theta)} = \frac{p(X|T,\Theta)p(T|\Theta)}{\int p\left(X,\hat{T}\middle|\Theta\right)p\left(\hat{T}\middle|\Theta\right)d\hat{T}}, \quad (2.2)$$

де X — деяке з значень спостережуваних змінних, а T— приховані змінні. Таким чином, нам необхідно знати тільки розподіл спостережуваної компоненти при фіксованій прихованої $p(X|T, \Theta)$ і імовірності прихованих даних $p(T|\Theta)$.

ЕМ-алгоритм ітеративно покращує початкову оцінку Θ_0, обчислюючи нові значення оцінок Θ_1,Θ_2 і так далі. На кожному кроці перехід від попереднього до наступного виконується таким чином:

$$Q(\Theta) = E_T[\log p(X,T|\Theta)|X] \quad (2.3)$$

де $Q(\Theta)$ — математичне очікування логарифма правдоподібності. Іншими словами, ми не можемо відразу обчислити точну правдоподібність, але за відомими даними X ми можемо знайти апостеріорну оцінку ймовірностей для різних значень прихованих змінних T .Для кожного набору значень T і параметрів Θ ми можемо обчислити математичне очікування функції правдоподібності з даного набору X. Воно залежить від попереднього значення Θ, бо це значення впливає на ймовірності прихованих змінних T.

$Q(\Theta)$ − обчислюється наступним чином:

$$\theta_{n+1} = \arg\max Q(\Theta) \quad (2.4)$$

іншими словами, умовне математичне сподівання $\log p(X,T|\Theta)$ при умові Θ.

Інакше кажучи, Θ_{n+1} – це значення, максимізуючи (М) умовне математичне сподівання (Е) логарифма правдоподібності при даних значеннях спостережуваних змінних і попередньому значенні параметрів.

За певних обставин зручно розглядати ЕМ-алгоритм як два кроки максимізації, що чергуються.

2.10 Алгоритм COBWEB

Алгоритм COBWEB – класичний метод інкрементальної кластеризації [12]. Він створює ієрархічну кластеризацію у вигляді дерева класифікації: кожен вузол цього дерева посилається на концепт і містить імовірнісний опис цього концепту, яке включає в себе ймовірність приналежності концепту до даного вузла і умовні ймовірності виду:

$$P\left(A_i = \nu_{ij} \middle| C_k\right), \tag{2.5}$$

де $A_i = \nu_{ij}$ – пара атрибут-значення, C_k - клас концепту.

Вузли, що знаходиться на певному рівні дерева класифікації, називають зрізом. Алгоритм використовує для побудови дерева класифікації евристичну міру оцінки, звану корисністю категорії - приріст очікуваного числа коректних припущень про значення атрибутів при знанні про їх належність до певної категорії щодо очікуваного числа коректних припущень про значення атрибутів без цього знання. Щоб додати новий об'єкт в дерево класифікації, алгоритм COBWEB ітеративно проходить все дерево в пошуках «кращого» вузла, до якого віднести цей об'єкт. Вибір вузла здійснюється на основі переміщення об'єкта в кожен вузол і обчислення корисності категорії отриманого зрізу. Також обчислюється корисність категорії для випадку, коли об'єкт відноситься до новостворюваного вузлу. У підсумку об'єкт відноситься до того вузла, для якого корисність категорії більше.

Недоліки:

- припускається, що розподіл імовірностей значень різних атрибутів статистично незалежні один від одного. Однак це припущення не завжди вірно, тому зазвичай між значеннями атрибутів існує кореляція;

– імовірнісний розподіл кластерів робить дуже складним їх оновлення, особливо в тому випадку, коли атрибути мають велике число можливих значень. Це викликано тим, що складність алгоритму залежить не тільки від кількості атрибутів, але і від кількості їхніх можливих значень.

2.11 Міри подібності

Для проведення кластеризації необхідно ввести поняття подібності об’єктів за їх властивостями. У кожний кластер повинні потрапити об’єкти, що мають подібні характеристики.

Ключовим моментом в кластерному аналізі даних вважається вибір метрики (або міри близькості об’єктів). Подібність або розходження між об’єктами класифікації встановлюється в залежності від обраної метричної відстані між ними. Якщо кожен об’єкт описується i властивостями (ознаками), то він може бути представлений як точка в i-мірному просторі, і схожість з іншими об’єктами буде визначатися як відповідна відстань. Розглянута задача кластеризації зводиться до задачі визначення функції близькості між об’єктами класів – вибору міри відстані між об’єктами.

Використовуються різні міри відстані. Розглянемо деякі з них

2.11.1 Евклідова відстань

Найбільш поширена відстань. Вона є географічною відстанню в багатовимірному просторі і обчислюється наступним чином:

$$d(X,Y) = \sqrt{\sum_{i=1}^{m}(X_i - Y_i)}\ , \qquad (2.6)$$

де $d(X, Y)$ – відстань між об’єктами X та Y; X_i – значення i-властивості об’єкта X; Y_i – значення i-властивості об’єкта Y.

З геометричної точки зору, евклідова міра відстані може виявитися некоректною, якщо ознаки виміряні в різних одиницях. Щоб виправити становище, вдаються до нормування кожної ознаки.

Застосування евклідової відстані у якості міри виправдано в наступних випадках:

- властивості (ознаки) об’єкта однорідні за фізичним змістом і однаково важливі для класифікації;
- простір ознак збігається з геометричним простором.

2.11.2 Квадрат евклідової відстані

Дана міра відстані використовується в тих випадках, коли потрібно надати більше значення більш віддаленим один від одного об’єктам. Це відстань обчислюється таким чином:

$$d(X,Y) = \sum_{i=1}^{m} (X_i - Y_i) \tag{2.7}$$

2.11.3 Зважена Евклідова відстань

Застосовується в тих випадках, коли кожній i-властивості вдається приписати певну «вагу», пропорційно ступеню важливості ознаки в задачі кластеризації

$$d(X,Y) = \sum_{i=1}^{m} w_i (X_i - Y_i), \tag{2.8}$$

де w_i – «вага» i-ї властивості. Визначення ваг, як правило, пов’язано з додатковими дослідженнями, наприклад, організацією опитування експертів і обробкою їх думок.

2.11.4 Хеммінгова відстань

Також називається Манхеттенською, або відстанню міських кварталів. Це відстань є різницею по координатах. У більшості випадків ця міра відстані призводить до таких же результатів, як і для звичайної відстані Евкліда. Проте вплив окремих великих різниць (викидів) зменшується (так як вони не зводяться в квадрат). Хеммінгова відстань обчислюється за формулою:

$$d(X,Y) = \sum_{i=1}^{m}(|X_i| - |Y_i|) \tag{2.9}$$

2.11.5 Відстань Чебишова

Цю відстань використовують, коли необхідно визначити два об’єкти як «різні», якщо вони різняться з якої-небудь однієї координати (яким-небудь одним виміром). Приймає значення найбільшого модуля різниці між значеннями відповідних властивостей (ознак) об’єктів:

$$d(X,Y) = \max\ |X_i - Y_i| \tag{2.10}$$

2.11.6 Показникова відстань

Для прогресивного збільшення або зменшення ваг, що відносяться до розмірності, для якої відповідні об’єкти сильно відрізняються. Це може бути досягнуто з використанням показникової відстані:

$$d(X,Y) = \left(\sum_{i=1}^{m}|X_i - Y_i|^p\right)^{1/p} \tag{2.11}$$

2.11.7 Відстань Канберра

Канберрова відстань лежить між 0 і 1, але є нечутливою до сильно асиметричних даних. Однак застосування цього методу при наявності нульових і негативних значень небажано, тому рекомендується попереднє перетворення даних.

Формула для обчислення відстані Кенберра:

$$d(X,Y) = \sum_{i=1}^{m} \frac{|X_i - Y_i|}{X_i + Y_i} \tag{2.12}$$

2.11.8 Відстань Брея-Картіса

Відстань Брея-Картіса має значення між 0 і 1, проте змінні з великими значеннями надають більший вплив на результат:

$$d(X,Y) = \frac{\sum_{i=1}^{m} |X_i - Y_i|}{\sum_{i=1}^{m} X_i + \sum_{i=1}^{m} Y_i} \tag{2.13}$$

Вибір міри відстані і ваг – дуже важливий етап, тому що від цих процедур залежать склад і кількість формованих класів, а також ступінь подібності об’єктів всередині класів.

Оцінка подібності між об’єктами сильно залежить від абсолютного значення ознаки й від ступеня його варіації в сукупності. Щоб усунути подібний вплив на процедуру класифікації, можна нормувати значення вихідних змінних.

Висновки до розділу

В розділі розглянуто та проаналізовано основні методи кластерного аналізу, на основі чого буде запропоновано спосіб вирішення задачі кластеризації адресів користувачів мереж блокчейну.

Метод k-середніх є найбільш простим, але в той же час задовольняє вимогам. Він розбиває безліч елементів векторного простору на заздалегідь відоме число кластерів k. Дія алгоритму таке, що він прагне мінімізувати середньоквадратичне відхилення на точках кожного кластера, що є необхідним для пошуку координат центру кластерів. Надалі, належність до кластеру для нових адрес можна буде визначити за відстанями до цих центрів.

В нашому випадку, останній недолік не є значимим, бо ми і так заздалегідь знаємо кількість кластерів та їх семантичне пояснення, необхідно лише визначити координати центрів цих кластерів.

Перший недолік нівелюємо тим, що застосуємо модифікацію алгоритму – k-means++. Суть її в тому, що ми випадковим чином визначає тільки перший центроїд, далі для кожної точки необхідно розрахувати квадратичну відстань до найближчого центроїду та вибираємо з цих точок наступний центроїд так, щоб ймовірність вибору точки була пропорційна обчисленому для неї квадрату відстані. Після цього можна застосовувати звичайний метод k-середніх. Модифікація була запропонована в 2007 році Девідом Артуром і Сергієм Вассильвітскім та є зараз найпоширенішою реалізацією методу k-середніх.

Щодо другого недоліку – він теж частково нівелюється застосуванням модифікації k-means++.

Розглянемо задачу кластеризації, як задачу мінімізації суми квадратів відстаней до центроїду від спостережуваних точок, обчислених методом найменших квадратів.

3 МЕТОДИ ВИЗНАЧЕННЯ ПАРАМЕТРІВ ТА КЛАСТЕРИЗАЦІЯ АДРЕС

3.1 Змістовна постановка задачі

В мережі блокчейну присутня множина блоків транзакцій, які пов’язані з множиною адрес. Необхідно за множиною транзакцій певної адреси встановити профіль (належність до кластеру) цієї адреси.

При аналізі адрес, використовуються наступні дані:

- кількість токенів у транзакції;
- час пасивності адреси;
- кількість токенів на «рахунку».

Для кожної з цих величин можна обрахувати математичне сподівання та дисперсію.

3.2 Математична постановка задачі

Нехай розглядається список адресів A, та для кожного адреса A_i список транзакцій T_i, довжиною n_i. Для кожної транзакції T_{ij} ($j \in 2..n_i$) можна визначити часовий інтервал t_{ij} з моменту проведення попередньої транзакції (ми не розглядаємо першу транзакцію, бо до неї дана адреса не проявляла активності).

Розглянемо наступні величини як випадкові для кожної адреси:

- кількість токенів у транзакції;
- час пасивності адреси;
- кількість токенів на «рахунку».

Для кожної з цих величин можна обрахувати математичне сподівання та дисперсію.

3.3 Метод визначення розподілу інтервалів часу пасивності адресів

Суть методу полягає у визначені характеристик для випадкової величини – часу пасивності адреси.

Час пасивності адреси t_{ij} – інтервал часу для адреси A_i між транзакціями T_{ij} та T_{ij-1}. Так як це є дискретною випадковою величиною, то для кожної адреси A_i та будь-якого інтервалу Δt можна визначити множину транзакцій з таким же часом пасивності:

$$f_i(\Delta t) = \{T_{ij} \mid t_{ij} = \Delta t\}, \tag{3.1}$$

де T_{ij} – j-та транзакція i-ої адреси ($j \in 2..n_i$), t_{ij} – час пасивності адреси T_{ij}. А ймовірність того, що для адреси A_i, транзакція буде мати часовий інтервал Δt визначається за формулою:

$$p_i(\Delta t) = \frac{|f_i(\Delta t)|}{n_i}, \tag{3.2}$$

де n_i – загальна кількість транзакцій для i-ої адреси. Тепер можна визначити математичне сподівання та дисперсію часу пасивності адреси A_i:

$$M_i[X] = \sum_{j=1}^{n_i} t_{ij} \cdot p_i(t_{ij}), \tag{3.3}$$

$$D_i[X] = \sum_{j=1}^{n_i} p_i(t_{ij}) \left(t_{ij} - M_i[X]\right)^2. \tag{3.4}$$

3.4 Метод визначення розподілу кількості токенів у адреси

Для кожної адреси A_i, кожна транзакція T_{ij} ($j \in 2..n_i$) має певну кількість токенів $v(T_{ij})$, що передаються або отримуються адресою A_i внаслідок транзакції T_{ij}.

Очевидно що $v(T_{ij}) < 0$, якщо внаслідок транзакції T_{ij}, адреса A_i віддала токени, та $v(T_{ij}) > 0$ – якщо отримала. Нехай $g(T_{ij})$ – множина транзакції адреси A_i, що відбулися до транзакції T_{ij}, включаючи саму T_{ij}. Тоді для кожної транзакції T_{ij} можна визначити кількість токенів у адреси A_i:

$$b_{ij} = \sum_{T_{ij} \in g(T_{ij})} v(T_{ij}). \tag{3.5}$$

Так як це є дискретною випадковою величиною, то для кожної адреси A_i та будь-якої кількості токенів *b,* можна визначити множину транзакцій з таким же значенням балансу:

$$f_i(b) = \{T_{ij} \mid b_{ij} = b\}. \tag{3.6}$$

А ймовірність того, що для адреси A_i, транзакція буде мати баланс b визначається за формулою:

$$p_i(b) = \frac{|f_i(b)|}{n_i}, \tag{3.7}$$

де $f_i(b)$ – множина транзакцій зі значенням балансу – b у адреси A_i, n_i – загальна кількість транзакцій для *i*-ої адреси. Тепер можна визначити математичне сподівання та дисперсію часу пасивності адреси A_i:

$$M_i[Y] = \sum_{j=1}^{n_i} b_{ij} \cdot p_i(b_{ij}), \tag{3.8}$$

$$D_i[Y] = \sum_{j=1}^{n_i} p_i(b_{ij}) \left(b_{ij} - M_i[Y]\right)^2, \tag{3.9}$$

де $p_i(b_{ij})$ – ймовірність того, *i*-ої адреси, транзакція буде мати баланс b_{ij}, n_i – загальна кількість транзакцій для *i*-ої адреси.

3.5 Визначення кількості кластерів

Для визначення кількості кластерів застосуємо метод ієрархічної кластеризації (розділ 2.5 Алгоритм ієрархічної кластеризації). Для визначення кількості кластерів, яка відображатиме різні моделі поведінки користувачів в певній предметній області, розглядатимемо відстань між кластерами та порівняння характеристик отриманих центроїдів. Кількість кластерів та їх характеристики можуть відрізнятись для різних мереж та різних предметних областей.

3.6 Метод визначення типу користувача за його поведінкою

Для визначення типу користувача за його поведінкою, застосуємо кластерний аналіз та метод k-середніх для визначеної в попередньому пункті кількості кластерів.

Насамперед необхідно визначити вектори, що характеризують адреси. В рамках кластерного аналізу, кожна адреса є вектором $(x_1, \dots, x_6)$, де x_1 – математичне сподівання часу пасивності, x_2 – дисперсія часу пасивності, x_3 – математичне сподівання кількості токенів у адреси, x_4 – дисперсія кількості токенів у адреси, x_5 – математичне сподівання кількості токенів в транзакції, x_6 – дисперсія кількості токенів в транзакції. Для визначення відстані між двома точками, використаємо формулу евклідової відстані:

$$d(p, q) = \sqrt{\sum_{k=1}^{n} (p_k - q_k)^2}, \tag{3.10}$$

де p та q – дві точки, $p = (p_1, \dots, p_n), q = (q_1, \dots, q_n)$, n – розмірність векторів.

Отже задача кластеризації зводиться до мінімізації суми квадратів відстаней між кожною точкою та центром її кластера, тобто:

$$V = \sum_{i=1}^{k} \sum_{x_j \in S_i} \left(x_j - \mu_i\right)^2, \tag{3.11}$$

$$V \to min \tag{3.12}$$

де V – сума квадратів відстаней між кожною точкою та центром її кластера, k – кількість кластерів, S_i – шукані кластери, μ_i – центр кластера S_i.

Висновки до розділу

В даному розділі був наведено змістовну та математичну постановки задачі кластеризації адрес в мережі блокчейну, наведено формальний опис предметної області. Наведено процедури визначення кількості кластерів та параметрів, які використовуються в задачі кластеризації. Наведено процедуру визначення характеристик кластерів.

4 ОПИС ПРОГРАМНОГО ПРОДУКТУ

4.1 Загальні вимоги до архітектури програмного забезпечення

Вимоги до архітектури програмного забезпечення складені з урахуванням потреби в побудові середовища для аналізу транзакцій, що буде незалежним від їх конкретних реалізацій, методів постачання або належність до певного ланцюжку блокчейн, та включатиме можливість використання різних методів аналізу цих транзакцій. Це призводить до потреби в винесенні вихідного коду вказаних частин в окремі компоненти, отже модульності системи.

Більше того, система повинна підтримувати можливість без зміни вихідного коду для аналізу мати можливість додавати нові ланцюжки блокчейну. Цього можна досягнути за рахунок загальних інтерфейсів для всіх класів, що дозволить їм взаємодіяти один з одним.

Також важлива вимога ефективності процесу розробки призвела до вибору платформи NodeJS та мови програмування JavaScript з використанням фреймворку Express.

JavaScript має низку властивостей об'єктно-орієнтованої мови, але завдяки концепції прототипів підтримка об'єктів в ній відрізняється від традиційних мов ООП. Крім того, JavaScript має ряд властивостей, притаманних функціональним мовам, — функції як об'єкти першого класу, об'єкти як списки, каррінг, анонімні функції, замикання (closures) — що додає мові додаткову гнучкість.

Express — програмний каркас розробки веб-застосунків для Node.js, реалізований як вільне і відкрите програмне забезпечення під ліцензією MIT. Він спроектований для створення веб-застосунків і API. Де-факто є стандартним каркасом для Node.js. Автор фреймворка, TJ Holowaychuk, описує його як створений на основі написаного на мові Ruby каркаса Sinatra, маючи на увазі, що він мінімалістичний, але має велику кількість плагінів, шо підключаються.

В якості інтегрованого середовища розробки обрано WebStorm від компанії JetBrains. Це є дуже зручним, оскільки вона повністю підтримує роботу з JavaScript та надає багато можливостей, які полегшують розробку.

4.2 Вимоги до технічного забезпечення

Структура технічних засобів визначається виходячи із можливості їх забезпечити виконання встановлених операцій процесу технічного обслуговування, можливості інтегрування до існуючих систем, захищеності від несанкціонованого доступу.

Для правильної роботи розробленого продукту до складу технічних засобів підприємства повинен входити комп'ютер, якій буде використовуватись в ролі серверу веб-додатку і має мати конфігурацію наведену нижче:

- процесор з тактовою частотою не нижче 1,6 ГГц;
- об'єм оперативної пам'яті не менше 8192 МБ;
- об'єм HDD або SSD не нижчий за 300Гб;
- доступ до мережі інтернет.

Для коректної роботи клієнтської частини необхідно комп'ютер, що задовольняє таким характеристикам:

- процесор з тактовою частотою не нижче 1,4 ГГц;
- об'єм оперативної пам'яті не менше 2048 МБ;
- доступ до мережі Інтернет.

Для роботи з додатком на кожному клієнтському робочому місці має бути встановлено один із таких веб-браузерів:

- Internet Explorer 10 або вище;
- браузери, що використовують движок Gecko 27.0 або вище;
- браузери, що використовують движок Trident 6.0 або вище;
- браузери, що використовують движок Webkit 537.31 або вище;
- браузери, що використовують движок Blink 537.11 або вище.

4.3 Підготовка вхідних даних

У зв’язку з тим, що програмні інтерфейси для компонентів програмного забезпечення є загальними для будь-якого блокчейну, але вихідний формат всередині мережі може відрізнятись, то необхідно привести ці дані до загальної структури. Розглянемо цю процедуру на прикладі блокчейну Bitcoin:

а) за допомогою програми Bitcoin Core завантажуємо весь блокчейн у внутрішньому форматі мережі. Перш за все, нас цікавить каталог blocks, який містить файли blk*.dat, кожен з яких містить інформацію для свого блоку у hex-форматі;

б) далі за допомогою скрипту, написаного на мові програмування Rust, дані з кожного файлу blk*.dat для кожного блоку переносимо до .csv файлів. Структура отриманих csv файлів наведена у таблицях 4.1-4.4. По закінченню цього кроку отримуємо наступні файли:

- blocks.csv;
- transactions.csv;
- tx_in.csv;
- tx_out.csv.

Таблиця 4.1 – Структура файлу blocks.csv

Назва поля	Тип поля	Опис поля
hash	string	хеш сума блоку
height	integer	порядковий номер блоку
version	integer	номер версії блоку, який вказує на набір правил підтвердження блоків, які потрібно дотримуватися
blocksize	integer	розмір даного блоку
hashPrev	string	хеш попереднього блоку

Продовження таблиці 4.1

hashMerkleRoot	string	merkle root hash походить від хешей всіх транзакцій, включених до цього блоку, що гарантує, що жоден з цих транзакцій не може бути змінений без зміни заголовка
nTime	long	час, коли майнер почав хешування
nBits	integer	розшифрована версія блоку повинна бути менше або дорівнювати цьому значенню
nNonce	integer	32-розрядне (4-байтове) поле, значення якого встановлено таким чином, що хеш блоку міститиме необхідну кількість нулів

Таблиця 4.2 – Структура файлу transactions.csv

Назва поля	Тип поля	Опис поля
txid	string	хеш транзакції
hashBlock	string	хеш блоку, до якого належить транзакція
version	integer	номер версії блоку, який вказує на набір правил підтвердження блоків, які потрібно дотримуватися
lockTime	integer	номер блоку, починаючи з якого, транзакція може бути додана до блокчейну

Таблиця 4.3 – Структура файлу tx_out.csv

Назва поля	Тип поля	Опис поля
txid	string	хеш транзакції
indexOut	integer	порядковий номер запису
value	long	кількість токенів, що відправляються
scriptPubKey	string	хеш відкритого ключа одержувача монет
address	string	адреса, на яку відсилаються токени
unspent	boolean	вказу є що вихід транзакції ще не є входом іншої транзакції

Таблиця 4.4 – Структура файлу tx_in.csv

Назва поля	Тип поля	Опис поля
txid	string	хеш транзакції
hashPrevOut	string	хеш попереднього виходу транзакції
indexPrevOut	integer	порядковий номер попереднього виходу транзакції
scriptSig	string	хеш відкритого ключа відправника монет

в) на останьому кроці, за відомою структурою даних та дампом бази даних у csv файлах, заповнюємо MySQL з метою більш зручного доступу до даних та можливістю виконання аналітичних запитів.

4.4 Структура компонент програмного забезпечення

На рисунку 4.1 зображений один із варіантів діаграми компонентів програмної системи. Всі інтерфейси для взаємодії описані в головному компоненті, на який посилаються інші компоненти.

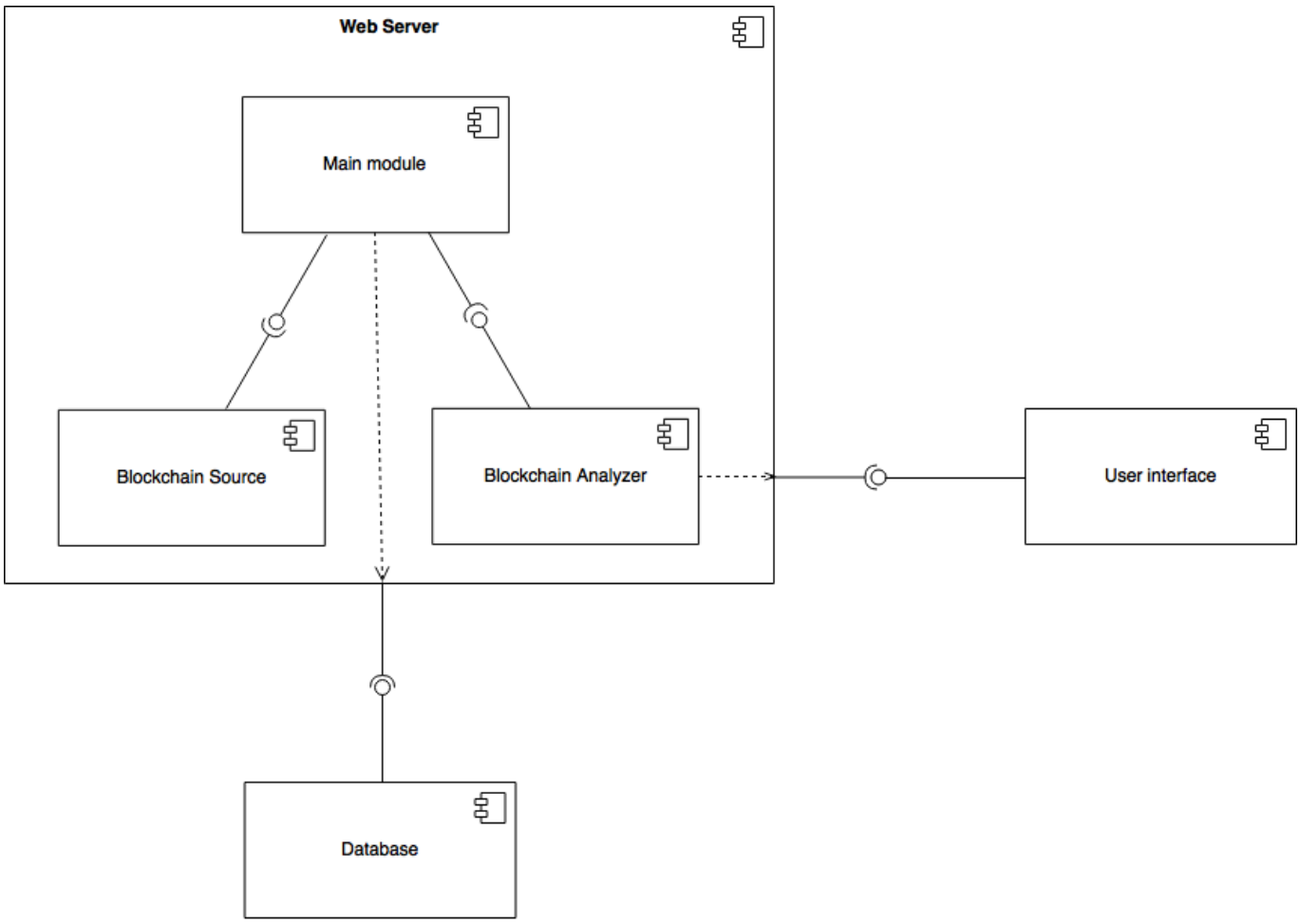

Рисунок 4.1 – Діаграма компонентів

Компонент надання даних передає всю необхідну інформацію до головного компонента, а він в свою чергу передає їх до компонента, що аналізує. Для того, щоб прискорити роботи з системою, було введено базу даних для кешування результатів аналізу. Коли компонент-аналізатор закінчив свою роботу, він показує користувачу результат, а головний компонент зберігає дані до бази та в наступний раз відобразить вже збережену інформацію.

4.5 Структура класів програмного забезпечення

Структура проекту включає в себе основний модуль, класи, що надають дані, та класи-аналізатори.

Всі класи, що надають дані, наслідуються від загального абстрактного та реалізують логіку асинхронного постачання даних про блокчейн згідно уніфікованого інтерфейсу. Кожен клас, що надає дані, працює з окремим блокчейном.

Клас-аналізатор також наслідується від загального абстрактного класу та отримує на вхід інформацію про блокчейн, виконує певні аналізуючі дії та асинхронно повертає результат.

Основний модуль забезпечує комунікацію між постачальником та аналізатором. При запуску програми, основний модуль «опитує» всі наявні класи, щодо того, з яким блокчейном вони працюють, або що саме вони аналізують. А після вибору користувачем пари «клас, що надає дані – аналізатор», зв’язує їх згідно інтерфейсу.

Діаграма класів наведена на рисунку 4.2:

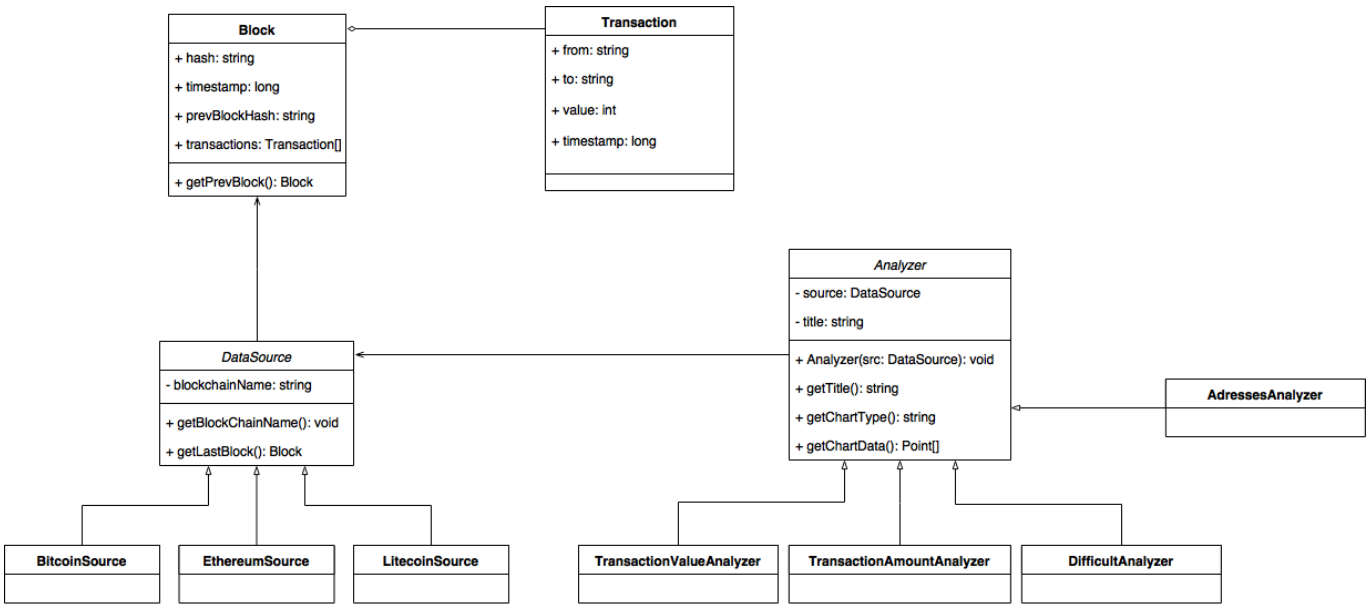

Рисунок 4.2 – Діаграма класів

У таблиці 4.5 наведено опис основних загальнодоступних функцій відповідно класів програми.

Таблиця 4.5 – Опис основних класів програми

Клас	Функція	Призначення
Block	getPrevBlock()	Для поточного блоку, повертає попередній
DataSource	getBlockChainName()	Повертає назву блокчейну з яким працює даний клас
DataSource	getLastBlock()	Повертає екзепляр класу Block, що є останнім у ланцюгу блокчейн
Analyzer	getTitle()	Повертає назву методу аналізу
Analyzer	getChartType()	Повертає назву одного з можливих варіантів візуалізації результатів аналізу
Analyzer	getChartData()	Повертає результат аналізу

4.6 Керівництво користувача

Розпочати роботу з Системою можливо перейшовши за посиланням . https://asoiudiploma.herokuapp.com.

Після успішного запуску Системи на екрані відкривається головне вікно Системи – екран «Ініціалізація аналізу» (рис. 4.3).

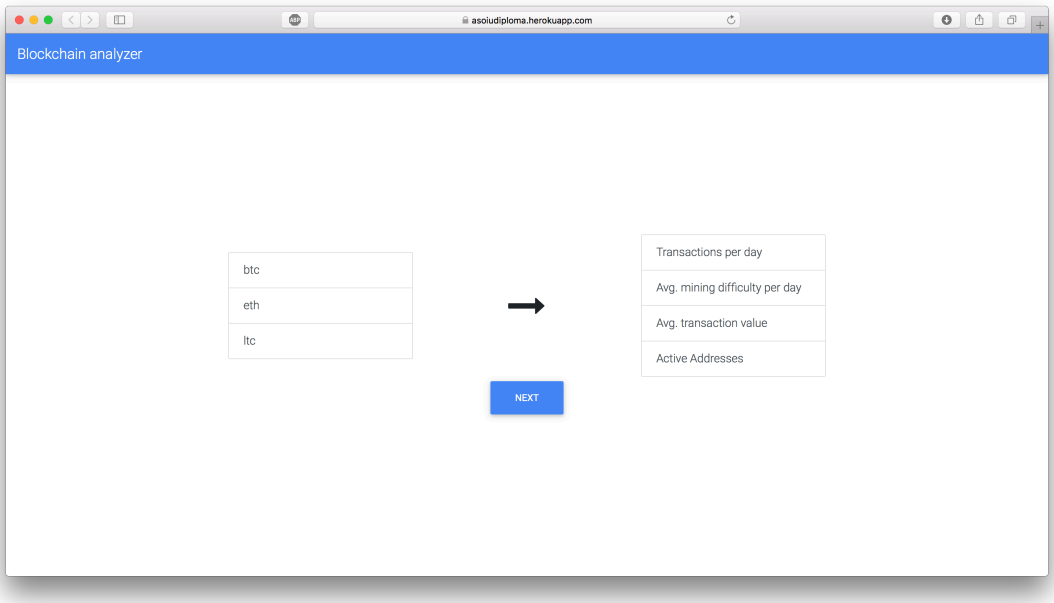

Рисунок 4.3 – Екран «Ініціалізація аналізу»

Головний екран містить список доступних в системі мереж блокчейну та список доступних методів аналізу. Після обрання одного з елементів з кожного списку, стає доступна кнопка далі. Далі, в залежності від обраного методу аналізу, відкриється певний екран для відображення результатів роботи методу. Один з таких типів таких результатів – графік, зображений на рисунку 4.4:

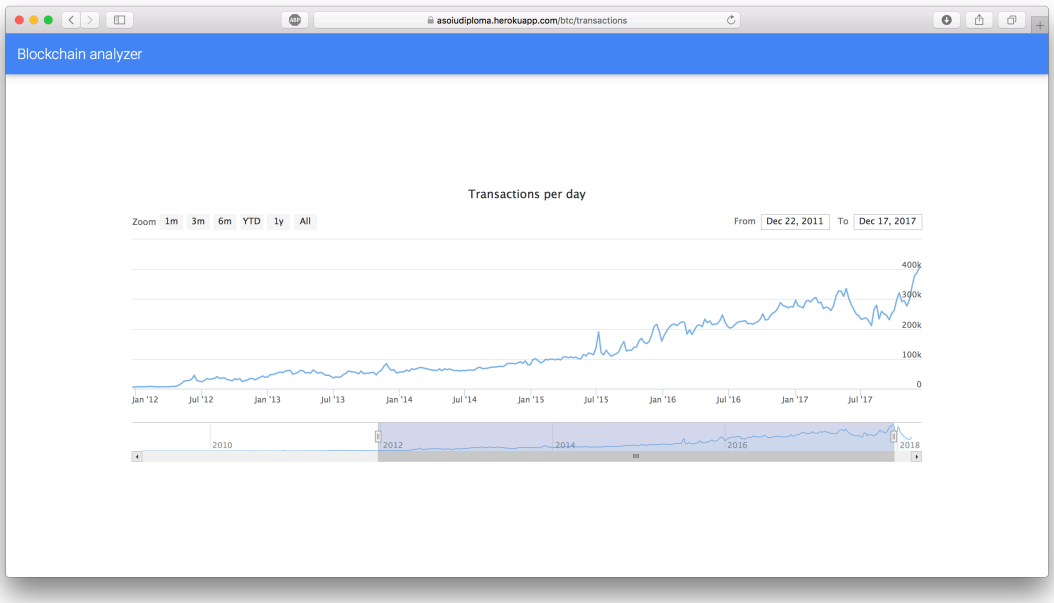

Рисунок 4.4 – Графік кількості транзакцій у день у мережі Bitcoin

Для перегляду детальної інформації за певною адресою, необхідно на головному екрані обрати пункт «Address clustering».

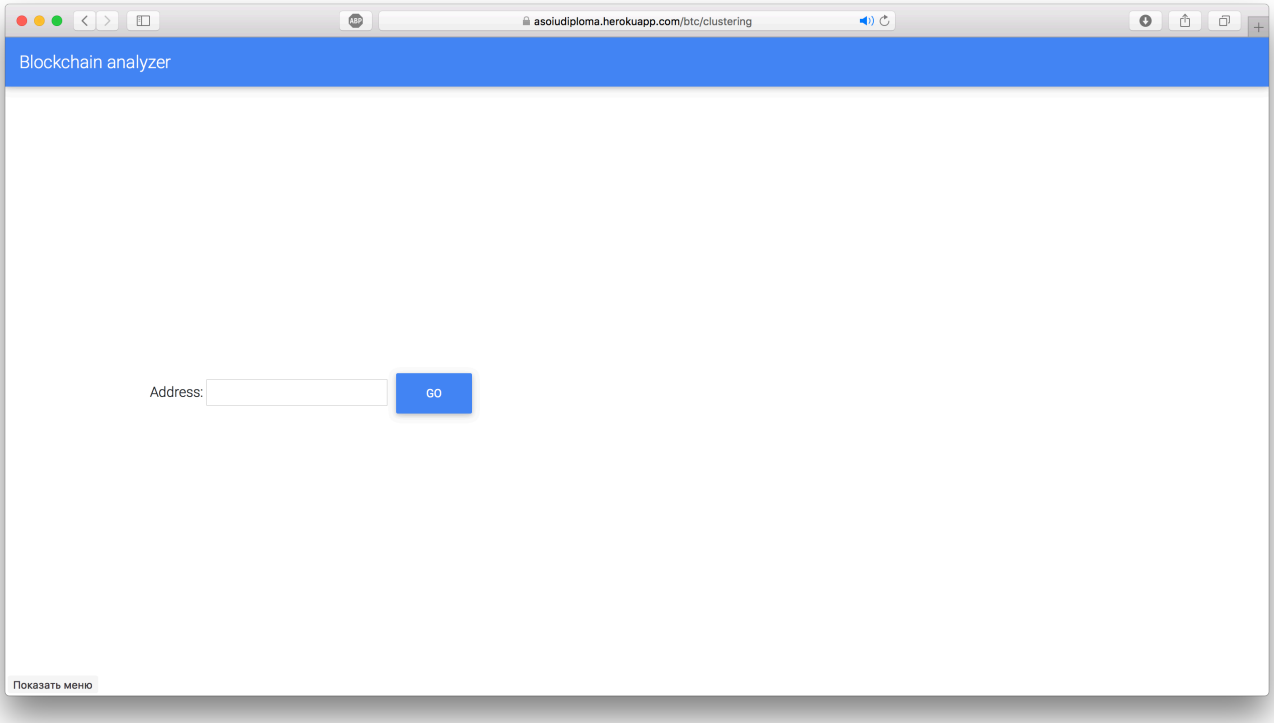

Рисунок 4.5 – Вікно введення адреси

Далі необхідно ввести шукану адресу, та натиснути кнопку “Go”.

Висновки до розділу

Програмно реалізовано алгоритми кластерного аналізу транзакцій ланцюжків блокчейн. Також розроблено та реалізовано архітектуру, що дозволяє працювати з кількома блокчейнами, не змінюючи реалізацію алгоритму аналізу. Програмне забезпечення дозволяє обрати блокчейн, ввести адресу та отримати інформацію щодо її профілю.

5 АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

5.1 Визначення кількості кластерів

При розробці програмного забезпечення необхідно заздалегідь визначити кількість кластерів. Для цього застосуємо ієрархічний метод кластеризації. Це дозволить визначити основні кластери та відстані між ними. Попередньо було підготовлено вхідні дані, обраховано необхідні параметри випадкових величин та нормалізовано їх значення. Для аналізу було обрано мережу Bitcoin та метод одиночного зв’язку(single linkage). Результати наведено на рисунку 5.1.

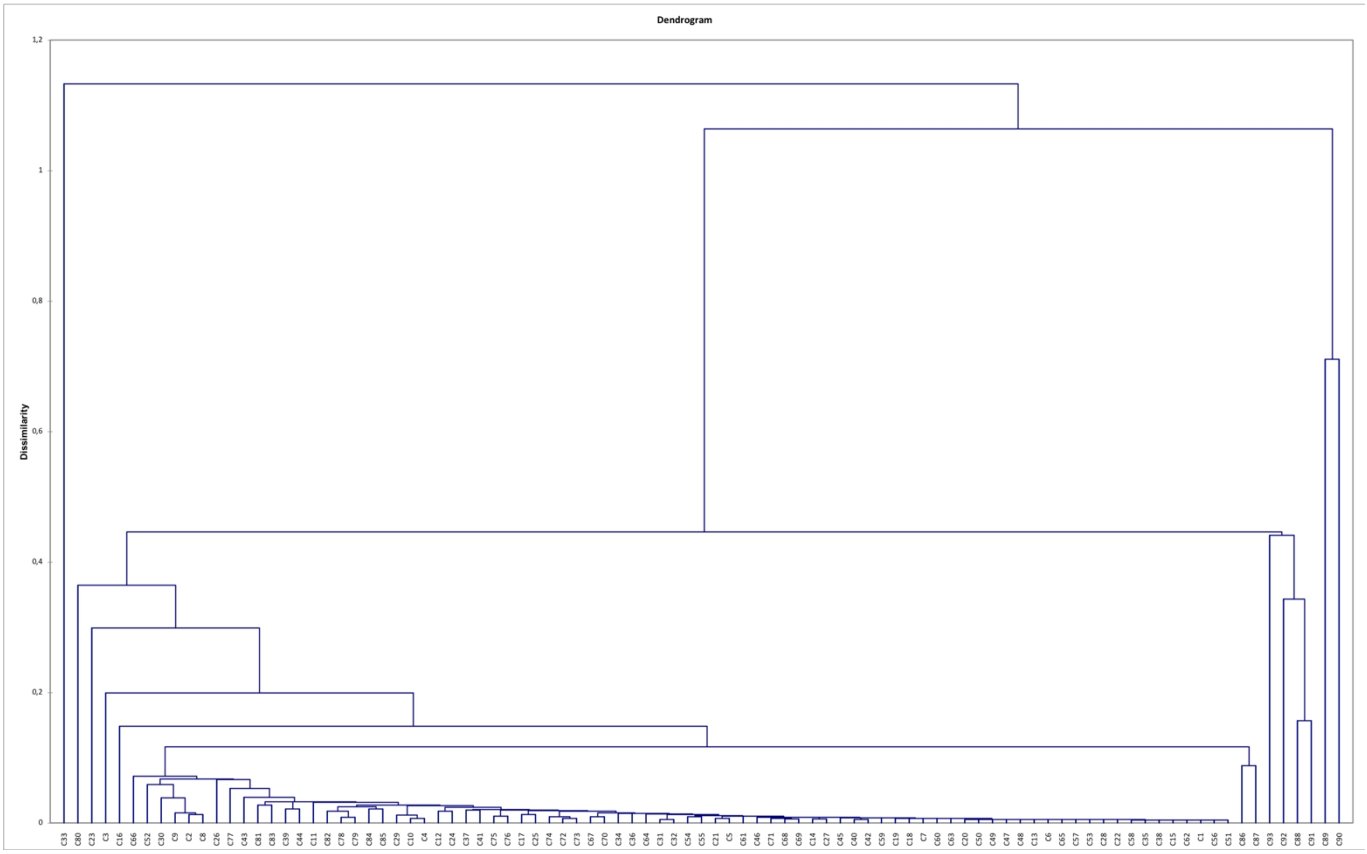

Рисунок 5.1 – Дендрограма методу одиночного зв'язку

Отже, необхідно дізнатись, при якому пороговому значенні об’єднання точок у кластери буде суперечити особливостям предметної області. Для цього знайдемо залежність розсіяння елементів у кластерах від кількості кластерів. Для цього для

кожного розбиття обрахуємо середню дисперсію кластерів. Графік цієї залежності наведено на рисунку 5.2.

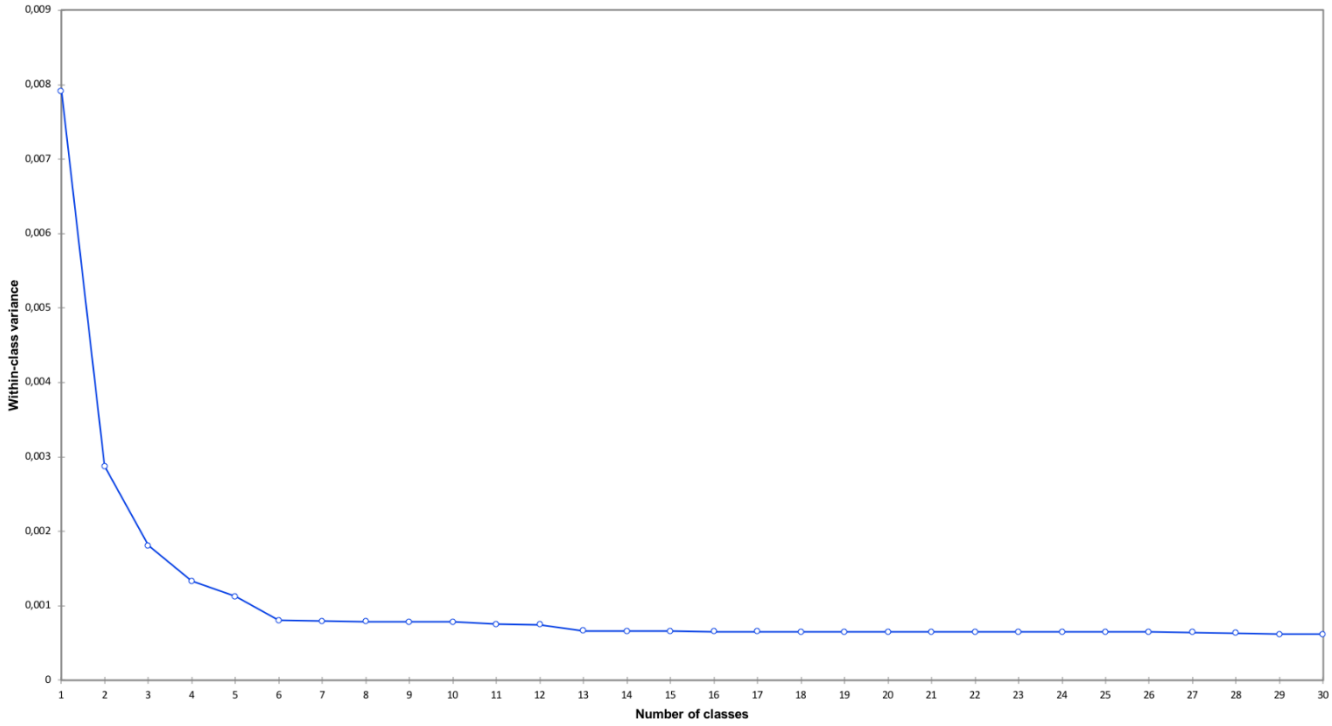

Рисунок 5.2 – Графік залежності середньої дисперсії кластера, від кількості кластерів

Розглянемо зміну величини дисперсії кластера в залежності від їхньої кількості. Очевидно, при зміні цієї величини менш ніж на 5% не варто виділяти новий кластер. Введемо порогову величину $\Delta_j = 1 - \frac{d_j}{d_{j-1}}$.

Таблиця 5.1 – Зміна величини дисперсії

j	Δ_j (%)
2	63,65806
3	37,20868
4	26,18233
5	15,60365
6	28,40694
7	1,586729
8	0,343342

Як бачимо з таблиці 5.1 при кількості кластерів 7 значення цієї величин не перевищує порогове значення у 5%. Отже для цієї предметної області недоцільно виділяти більше 6 кластерів. Тож розглянемо характеристики кластерів для розбиття адресів на шість кластерів. Віддаленість кластерів один від одного наведені в таблиці 5.2.

Таблиця 5.2 – Матриця відстаней між центроїдами

	1	2	3	4	5	6
1	0	0,199	0,593	0,394	0,891	1,930
2	0,199	0	0,394	0,213	0,693	1,830
3	0,593	0,394	0	0,199	0,299	1,678
4	0,394	0,213	0,199	0	0,498	1,746
5	0,891	0,693	0,299	0,498	0	1,615
6	1,930	1,830	1,678	1,746	1,615	0

Як бачимо, найближчими є пари кластерів 1, 2 та 3, 4. Розглянемо характеристики їх центроїдів.

Таблиця 5.3 – Нормалізовані центроїди кластерів

	1	2	3	4
Математичне сподівання кількості токенів у транзакції	0,00488	0,20197	0,59339	0,39558
Дисперсія кількості токенів у транзакції	0,00035	0,00000	0,00000	0,00000
Математичне сподівання часу пасивності адреси	0,02243	0,01097	0,00002	0,00070

Продовження таблиці 5.3

Дисперсія часу пасивності адреси	0,00094	0,00033	0,00000	0,00000
Математичне сподівання кількості токенів на «рахунку»	0,00166	0,02329	0,06844	0,04562
Дисперсія кількості токенів на «рахунку»	0,00004	0,00106	0,00900	0,00400

Як бачимо, кластери 1 та 2 мають найменші показники і різниця між ними не є значною. Тож їх можна поєднати і проаналізувати розподіл з пятьма кластерами.

Таблиця 5.4 – Матриця відстаней для розбиття з п’ятьома кластерами

	1	2	3	4	5
1	0	0,591	0,392	0,890	1,929
2	0,591	0	0,199	0,299	1,678
3	0,392	0,199	0	0,498	1,746
4	0,890	0,299	0,498	0	1,615
5	1,929	1,678	1,746	1,615	0

З матриці відстаней можна бачити, що кластери 2 та 3 теж є доволі близькими відносно інших. Також варто зауважити, що відстані між кластерами 2-5 не змінились після об’єднання кластерів на попередньому кроці. Це свідчить про те, що не відбулось перерозподілу елементів між кластерами. Розглянемо характеристики центроїдів другого та третього кластерів:

Таблиця 5.5 – Нормалізовані центроїди кластерів

	2	3
Математичне сподівання кількості токенів у транзакції	0,59339	0,39558
Дисперсія кількості токенів у транзакції	0,00000	0,00000
Математичне сподівання часу пасивності адреси	0,00002	0,00070
Дисперсія часу пасивності адреси	0,00000	0,00000
Математичне сподівання кількості токенів на «рахунку»	0,06844	0,04562
Дисперсія кількості токенів на «рахунку»	0,00900	0,00400

Як бачимо, характеристи в них дуже схожі, але до другого кластеру належать адреси з більшою кількістю токенів, що роблять більші транзакції. Зважаючи на невеликий час пасивності, вероґідно це кластер «торговців», що розбитий на «багатих» та більш «дрібних». Тож поєднаємо їх та розглянемо відстані між новими кластерами.

Таблиця 5.6 – Матриця відстаней для розбиття на чотири кластери

	1	2	3	4
1	0	0,591	0,889	1,929
2	0,591	0	0,299	1,678
3	0,889	0,299	0	1,615
4	1,929	1,678	1,615	0

З результату бачимо наявність двох кластерів, що є найближчими, але найменша відстань є в декілька разів більшою, ніж на попередніх кроках.

Таблиця 5.7 – Нормалізовані центроїди кластерів

	2	3
Математичне сподівання кількості токенів у транзакції	0,59339	0,89011
Дисперсія кількості токенів у транзакції	0,00000	0,00000
Математичне сподівання часу пасивності адреси	0,00002	0,00255
Дисперсія часу пасивності адреси	0,00000	0,00000
Математичне сподівання кількості токенів на «рахунку»	0,06844	0,10266
Дисперсія кількості токенів на «рахунку»	0,00900	0,02024

Бачимо, що характеристики центроїдів значно розрізняються, що свідчить про те, що надалі немає сенсу об’єдувати кластери. Але все ж переглянемо, як зміняться відстані, якщо їх поєднати:

Таблиця 5.8 – Матриця відстаней при розбитті на три кластери

	1	2	3
1	0	0,888	1,928
2	0,888	0	1,615
3	1,928	1,615	0

Як бачимо, відстані збільшились майже в три рази, що свідчить про те, що далі об’єднувати кластери не можна з точки зору предметної області та доцільно виділити чотири кластери, представники яких мають такі характеристики:

- майнер – користувач, що витрачає свої обчислювальні ресурси на забезпечення роботи мережі та створення нових блоків. Характеризується більшою кількістю токенів, отриманих з coinbase. В певних блокчейнах може не існувати;
- торговець – користувач, що не створює нові блоки, але активно приймає участь у транзакціях. Характеризується малим інтервалом пасивності;
- ходлер – користувач, що не дуже активно робить транзакції, як правило, просто накопичує токени у себе на рахунку. Характеризується великим інтервалом пасивності та великою середньою кількістю токенів у транзакції;
- загублена адреса – користувач, що в певний момент часу перестав проявляти будь-яку активність в мережі.

5.2 Аналіз мережі Bitcoin

Алгоритм було застосовано на даних мережі Bitcoin. Заздалегідь, для адрес було обраховано та нормалізовано необхідні вхідні дані, а саме:

- математичне сподівання та дисперсія кількості токенів у транзакції;
- математичне сподівання та дисперсія часу пасивності адреси;
- математичне сподівання та дисперсія кількості токенів на «рахунку»;
- кількість транзакцій, що надійшли до адреси внаслідок майнінгу.

Варто зазначити, що не у кожній мережі блокчейн можливо визначити майнерів та кількість транзакцій, що надійшли до адреси внаслідок майнінгу. Існують блокчейни, наприклад Ripple, в яких заздалегідь було створену певну кількість токенів, та нові не створюються з часом.

Після проведення аналізу та денормалізації, були отримані координати центроїдів кожного кластеру (таблиця 5.9):

Таблиця 5.9 – Координати центроїдів кластерів в мережі Bitcoin

	Ходлер	Майнер	Торговець	Загублена адреса
Математичне сподівання кількості токенів у транзакції	0.13	0.24	0.43	0.19
Дисперсія кількості токенів у транзакції	0.067	0.3	0.16	0.05
Математичне сподівання часу пасивності адреси	4 дні	7 днів	13 годин	13 днів 12 годин
Дисперсія часу пасивності адреси ($с^2$)	1028069275512	2890111288882	78951625366	7979246962890
Математичне сподівання кількості токенів на «рахунку»	0.25	0.28	0.36	0.27
Дисперсія кількості токенів на «рахунку»	0.13	0.34	0.61	0.13
Кількість транзакцій з coinbase	3.74	0.62	0.1	0.02

Отримавши вектори, що визначають позиції центроїдів, можна перейти до обчислення середньої віддаленості елементів всередині кожного кластеру, за формулою (2.8):

– ходлер – 0.22;
– майнер – 0.24;
– торговець – 0.27;
– загублена адреса – 0.15.

Розподіл кластерів за кількістю адресів зображено на рисунку 5.3:

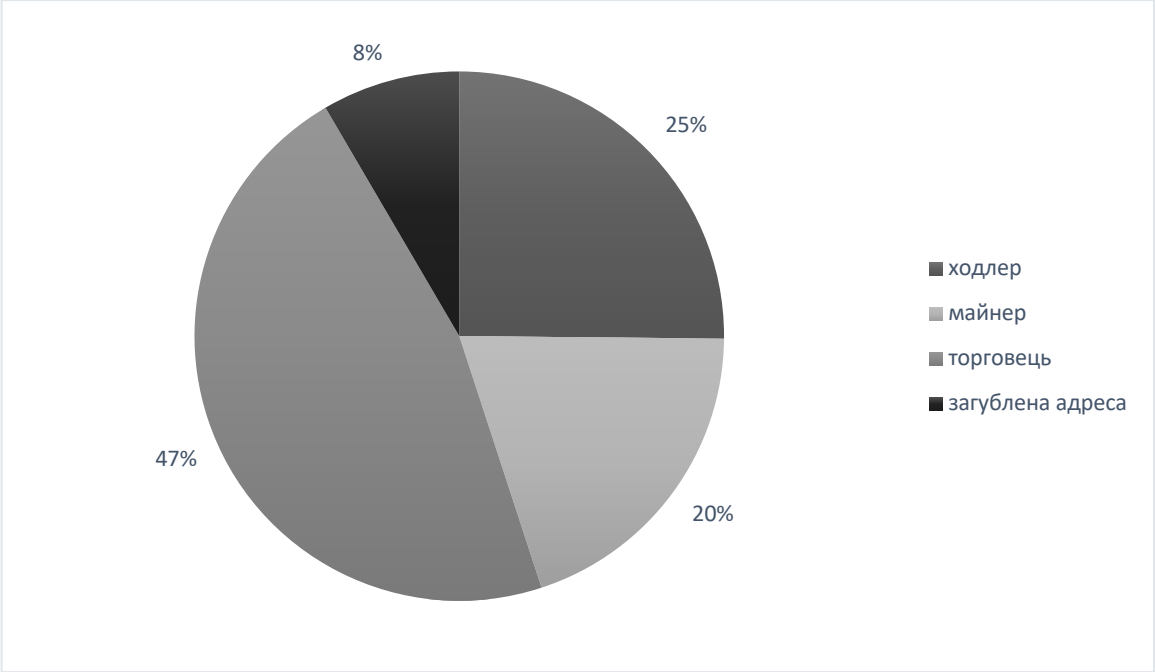

Рисунок 5.3 – Розподіл кластерів за кількістю адрес в мережі Bitcoin

5.3 Аналіз мережі Ethereum

Аналогічно до попереднього розділу, застосуємо алгоритм на мережі Ethereum. Результати аналізу наведені у таблиці 5.10.

Таблиця 5.10 – Координати центроїдів кластерів в мережі Ethereum

	Ходлер	Майнер	Торговець	Загублена адреса
Математичне сподівання кількості токенів у транзакції	0,18	0,17	0,51	0,24
Дисперсія кількості токенів у транзакції	0,09	0,33	0,22	0,05
Математичне сподівання часу пасивності адреси	2 дні 20 годин	7 днів	18 годин	10 днів
Дисперсія часу пасивності адреси ($с^2$)	784231791552	3052957330480	104724898792	9306133507382
Математичне сподівання кількості токенів на «рахунку»	0,3	0,39	0,48	0,35
Дисперсія кількості токенів на «рахунку»	0,14	0,43	0,54	0,16
Кількість транзакцій з coinbase	4,22	0,48	0,11	0,02

За отриманими векторами, обрахуємо середню віддаленість елементів всередині кожного кластеру, за формулою (2.8):

- ходлер – 0.29;
- майнер – 0.2;
- торговець – 0.36;
- загублена адреса – 0.12.

Розподіл кластерів за кількістю адресів зображено на рисунку 5.4:

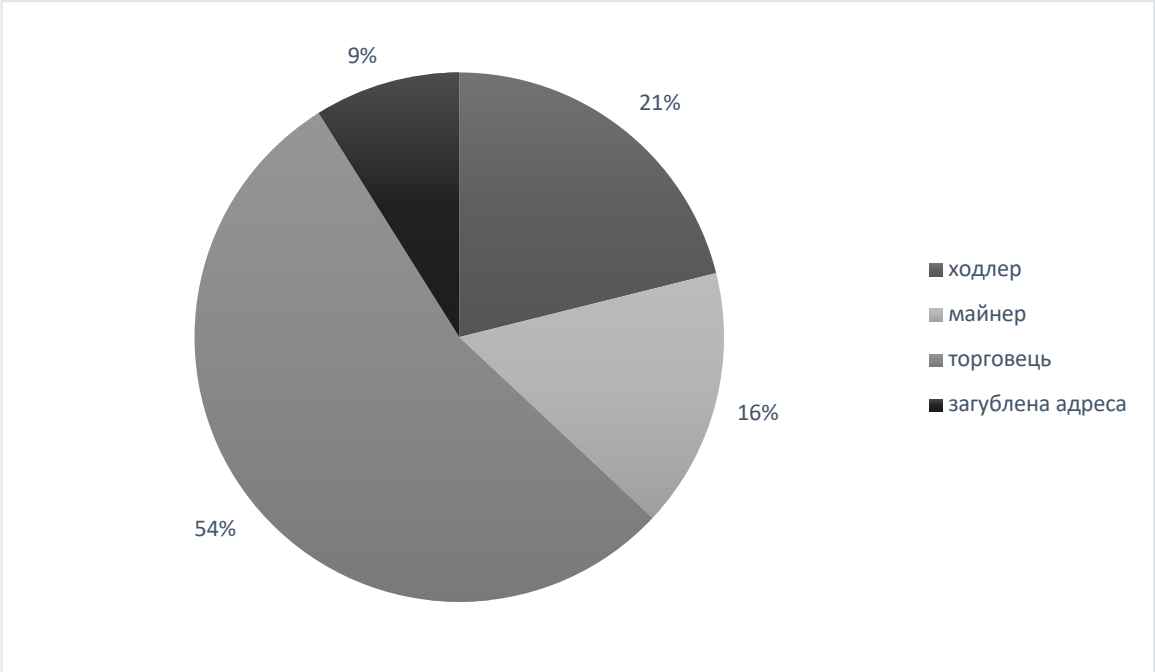

Рисунок 5.4 – Розподіл кластерів за кількістю адрес в мережі Ethereum

5.4 Задача аналізу тенденції зміни розмірів кластерів

Прикладом застосування даного алгоритму є спостереження за зміною розмірів кластерів та їх співвідношенням. Аналізуючи тенденцію змін розмірів кластерів, можна передбачити майбутню зацікавленість користувачів у певній мережі, або навпаки – відтік їх активної частини. Наприклад, якщо спостерігається швидке зменшення кількості загублених адресів та збільшення торговців, то це може свідчити про різке зростання зацікавленості в певному блокчейні, як наслідок – зростання цінності токенів та збільшення кількості майнерів, що призводить но збільшення хешрейту мережі та її складності майнінгу.

5.5 Задача аналізу поведінки певних користувачів

Також, прикладом застосування є задача виявлення ролі користувача за його адресою. Отримавши з мережі список транзакцій по адресі та обрахувавши значення вхідних даних для алгоритму, можна обчислити відстані до отриманих раніше центроїдів. Найменша відстань вказує на належність адреси до певного кластеру. Це дозволить визначити профіль користувача, попередити про можливу передачу

токенів до загубленої адреси, показати активність користувача, що може попередити дії шахраїв, тощо.

Висновки до розділу

В даному розділі було визначено кількість кластерів для мереж Bitcoin та Ethereum, проведено їх кластерний аналіз та визначено характеристики отриманих кластерів. За результатами аналізу для мереж Bitcoin та Ethereum визначено склад цих мереж: приблизно половина адрес є активними учасниками, а п’ята частина – майнери, що забезпечують функціонування мережі. Також можна помітити більший середній розмір транзакцій у мережі Ethereum, одна з причин – менша цінність токенів у порівнянні з Bitcoin, але водночас – достатній рівень зацікавленості з боку користувачів.

ВИСНОВКИ

У даній магістерській дисертаціїї було розглянуто поширені методи аналізу та оцінки ланцюжків тразнакцій в мережах блокчейну, проведено їх порівняльний аналіз. Було реалізовано алгоритм та створено програмний засіб для кластеризації та оцінки адресів мереж блокчейн. Програмна реалізація системи написана мовою програмування JavaScript на платформі NodeJS з використанням фреймворку ExpressJS. Такий вибір зроблено на користь використання розвиненого середовища розробки WebStorm та спрощеня розробки інтерфейсу користувача через використання веб-технологій та мов програмування HTML, CSS та JavaScript, зокрема варто відмітити наявність зручних статистичних функцій та функцій створення різноманітних графіків і інших методів, що дозволяють задовольнити вимогу створення зручного графічного інтерфейсу користувача.

Першим етапом розробки цього проекту був аналіз предметної області. Були досліджені останні теоретичні роботи у даній сфері, виокремлено та використано в роботі результати найбільш перспективних із них.

На основі проведених досліджень стало можливим зробити обґрунтований вибір найбільш ефективного для даної задачі алгоритму і оптимальних значень його настроюваних параметрів. Був розроблений алгоритм для кластеризації та архітектура, що дозволить застосувати його для різних реалізацій та мереж блокчейну.

В рамках опису архітектури побудованої системи розглянуто перелік усіх її модулів, наведено діаграми компонентів, що можуть бути утворені в результаті динамічного зв'язування компонент. У якості документації дана інструкція користувача з ілюстраціями.

Таким чином в результаті проведеної роботи була досягнута поставлена мета дослідження.

ПЕРЕЛІК ПОСИЛАНЬ

1. 12 способов определить состояние Биткойна [Електроний ресурс] / Режим доступу: https://bitnovosti.com/2014/10/17/12-metrik-bitcoina/

2. Coinometrics [Електроний ресурс] / Режим доступу: http://www.coinometrics.com

3. Bitcoin Block Explorer - Blockchain [Електроний ресурс] / Режим доступу: https://blockchain.info

4. Задача кластеризації адрес в мережі блокчейн / Данильчук Р. К., Жураковська О. С. // Міжнародний науковий журнал "Інтернаука". — 2018. — №9.

5. Dmitry Ermilov / Automatic Bitcoin Address Clustering // Machine Learning and Applications (ICMLA), 2017 – №16.

6. Harry Kalodner. BlockSci: Design and applications of a blockchain analysis platform [Електронний ресурс] / Режим доступу: https://cseweb.ucsd.edu/~smeiklejohn/files/imc13.pdf

7. Sarah Meiklejohn. A Fistful of Bitcoins: Characterizing Payments Among Men with No Names [Електронний ресурс] / Режим доступу: https://arxiv.org/pdf/1709.02489.pdf

8. Кластерний аналіз [Електронний ресурс] / Режим доступу: https://uk.wikipedia.org/wiki/Кластерний_аналіз

9. Кластерний аналіз [Електронний ресурс] / Режим доступу: https://uk.wikipedia.org/wiki/Кластерний_аналіз

10. Завдання кластеризації — Студопедія [Електронний ресурс] / Режим доступу: http://studopedia.com.ua/1_14742_zavdannya-klasterizatsii.html

11. Аналіз алгоритмів кластеризації для задач інтелектуального аналізу даних / Ю. В. Волосюк // Збірник наукових праць Військового інституту Київського національного університету імені Тараса Шевченка. - 2014. - Вип. 47. - С. 112-119. - Режим доступу: http://nbuv.gov.ua/UJRN/Znpviknu_2014_47_19

12. Cobweb (clustering) [Електронний ресурс] / Режим доступу: https://en.wikipedia.org/wiki/Cobweb_(clustering)

13. Факторный, дискриминантный и кластерный анализ: Пер. Ф18 с англ./Дж.-О. Ким, Ч. У. Мьюллер, У. Р. Клекка и др.; Под ред. И.С. Енюкова. – М.: Финансы и Статистика, 1989 – 215 с.

14. Аналіз основних принципів технології blockchain / Р. К. Данильчук, О. С. Жураковська. // Науковий Огляд. – 2017. – №10 (42). – С. 54–64.

15. D. Ron and A. Shamir (2012). Quantitative analysis of the full bitcoin transaction graph. [Електроний ресурс] / Режим доступу: http://eprint.iacr.org/2012/584

16. Satoshi Nakamoto (2008). Bitcoin: a peer-to-peer electronic cash system URL: https://bitcoin.org/bitcoin.pdf

17. Кластеризація методом k-середніх [Електронний ресурс] / Режим доступу: https://uk.wikipedia.org/wiki/Кластеризація_методом_к–середніх

18. S. Meiklejohn et al. «A fistful of Bitcoins: characterizing payments among men with no names» Proceedings of the 2013 conference on Internet measurement conference, pp. 127-140. ACM, 2013.

19. A.M. Antonopoulos. Mastering Bitcoin: unlocking digital cryptocur- rencies. OReilly Media, 2014.

20. E. Ben-Sasson et al. (2014). Zerocash: decentralized anonymous payments from Bitcoin (extended version). platform [Електронний ресурс] / Режим доступу: https://eprint.iacr.org/2014/349

21. Docs | Node.js [Електронний ресурс] / Режим доступу: https://nodejs.org/en/docs/

22. Express 4.x - API Reference [Електронний ресурс] / Режим доступу: http://expressjs.com/en/4x/api.html

23. Наукова конференція IOT-2018. [Електронний ресурс] // Режим доступу: http://asu.kpi.ua/naukova-konferentsiya-iot-2018/ (дата звернення: 28.04.2018)

24. Справочник по веб-технологиям | MDN [Електронний ресурс] / Режим доступу: https://developer.mozilla.org/ru/docs/Web/Reference

25. MongoDB Documentation [Електронний ресурс] / Режим доступу: https://docs.mongodb.com

26. Cluster analysis - Wikipedia [Електроний ресурс] / Режим доступу: https://en.wikipedia.org/wiki/Cluster_analysis

27. Hierarchical clustering - Wikipedia [Електроний ресурс] / Режим доступу: https://en.wikipedia.org/wiki/Hierarchical_clustering

28. k-means clustering - Wikipedia [Електроний ресурс] / Режим доступу: https://en.wikipedia.org/wiki/K-means_clustering

29. Jonathan Drake. Accelerated k-means with adaptive distance bounds [Електроний ресурс] / Режим доступу: http://opt.kyb.tuebingen.mpg.de/papers/opt2012_paper_13.pdf

30. Дюран Б. Кластерный анализ / Б. Дюран, П. Оддел., 2012. – 128 с.

31. Highcharts JS API Reference [Електроний ресурс] / Режим доступу: https://api.highcharts.com/highcharts/

32. Сложность. Все о криптовалюте - Bitcoin Wiki [Електроний ресурс] / Режим доступу: https://ru.bitcoinwiki.org/wiki/Сложность/

33. What is Bitcoin? – Blockchain Support Center [Електроний ресурс] / Режим доступу: https://support.blockchain.com/hc/en-us/articles/211122603-What-is-Bitcoin/

34. What is Ethereum? A Step-by-Step Beginners Guide [Ultimate Guide] [Електроний ресурс] / Режим доступу: https://support.blockchain.com/hc/en-us/articles/211122603-What-is-Bitcoin/

35. Programmable blockchains in context: Ethereum's future, by Vinay Gupta [Електроний ресурс] / Режим доступу: https://medium.com/humanizing-the-singularity/by-the-end-of-this-article-youre-going-to-understand-blockchains-in-general-and-ethereum-a-next-e11df6a1d7cf

36. Here's The Fundamental Bitcoin Valuation Metric That Determines The Cryptocurrency's Price Moves [Електроний ресурс] / Режим доступу: https://www.forbes.com/sites/iowa/2018/03/01/one-way-to-build-a-startup-hub-reduce-the-cost-of-prototyping/#ca4ab24166c8

ДОДАТОК А ГРАФІЧНІ МАТЕРІАЛИ

ПЛАКАТ 1 СХЕМА СТРУКТУРНА КЛАСІВ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПЛАКАТ 2 СХЕМА СТРУКТУРНА КОМПОНЕНТІВ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПЛАКАТ 3 ДЕНДРОГРАМА КЛАСТЕРІВ МЕРЕЖІ

ПЛАКАТ 4 ГРАФІК ЗАЛЕЖНОСТІ ДИСПЕРСІЇ КЛАСТЕРІВ ВІД КІЛЬКОСТІ КЛАСТЕРІВ

ПЛАКАТ 5 ВИЗНАЧЕННЯ КІЛЬКОСТІ КЛАСТЕРІВ В МЕРЕЖІ BITCOIN

ПЛАКАТ 6 ДЕНОРМАЛІЗОВАНІ КООРДИНАТИ ЦЕНТРОЇДІВ МЕРЕЖІ BITCOIN

ПЛАКАТ 7 ДЕНОРМАЛІЗОВАНІ КООРДИНАТИ ЦЕНТРОЇДІВ МЕРЕЖІ ETHEREUM